Trabajo Fin de Máster
Ingeniería de Telecomunicación

Implementación de Asignación Jerárquica Latente de Dirichlet para Modelado de Temas

Autor: María Silvestre Gómez

Tutor: Juan José Murillo Fuentes

Dep. Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2018

Trabajo Fin de Máster
Ingeniería de Telecomunicación

Implementación de Asignación Jerárquica Latente de Dirichlet para Modelado de Temas

Autor:
María Silvestre Gómez

Tutor:
Juan José Murillo Fuentes
Profesor Catedrático de Universidad

Dep. de Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2018

Trabajo Fin de Máster: Implementación de Asignación Jerárquica Latente de Dirichlet para Modelado de Temas

Autor: María Silvestre Gómez

Tutor: Juan José Murillo Fuentes

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2018

El Secretario del Tribunal

A mi familia

Agradecimientos

A mi abuela porque siempre estarás conmigo y por enseñarme a ser una mejor persona. Espero que estés orgullosa. A mi madre por el esfuerzo realizado todos estos años, por estar a mi lado y por ayudarme a crecer. A mi hermana por las aventuras compartidas, incluso las más pequeñas, y las que nos quedan por vivir. A mis tíos, Cinta y Jesús, y mis dos pequeños, Rubén y Manuel, por alegrarme todos los días y por estar en los buenos y en los malos momentos. A todos ellos, gracias por creer en mí.

A mi tutor, Juan José Murillo, por ofrecerme la oportunidad de realizar este proyecto y permitirme profundizar en este área de conocimiento fascinante. A él, junto al resto de profesores que me han acompañado a lo largo de estos años, agradecer todo lo que he aprendido de cada uno de ellos.

A todos los que siempre estuvieron a mi lado dándome su apoyo incondicional y a las maravillosas personas con las que he coincidido durante estos años universitarios, con los que he compartido grandes historias y junto a los que he crecido, gracias. A las todas las personas que he conocido en este último año, de los que he aprendido tanto profesional como personalmente y de las que espero seguir aprendiendo, gracias por enseñarme con tanta paciencia y por la confianza depositada en mí. Y en especial, a Jesús, por su ayuda, motivación cuando lo necesitaba y por alegrarme en los días de estrés.

María Silvestre Gómez

Sevilla, 2018

Resumen

Desde hace unos años, las técnicas de aprendizaje automático han ido evolucionando hasta alcanzar una gran importancia en la vida diaria. En la era de la tecnología en la que se generan gran cantidad de datos constantemente, se hace necesario el uso de técnicas automáticas que permitan el procesado de dichos datos y la extracción de la información para elaborar conclusiones. Por ejemplo, la información que se puede extraer de las redes sociales resulta de gran interés para las campañas de marketing, facilitando el acceso a un gran público y focalizando dichas campañas al sector que pueda estar interesado en ellas.

Un campo de actuación del aprendizaje automático es la minería de datos que se encarga de extraer y procesar información de un conjunto de textos: libros, reseñas, artículos, tweets, … y dentro de la minería de datos, se encuentran las técnicas de topic modeling, que se encargan de obtener los temas presentes es los textos. Hay un gran número de algoritmos que se encargan de resolver este problema atendiendo a diversas fuentes de información.

Este proyecto se centrará la investigación de las técnicas empleadas en la resolución de problemas de topic modeling presentes en el estado del arte y en la elección y desarrollo de una de éstas, HLDA, que elabora un árbol jerárquico de temas y cada documento seguirá una de las ramas de este árbol y los temas de los que trata se corresponderán con cada uno de los nodos de dicha rama. Dicho algoritmo empleará modelos complejos como son el proceso de restaurante chino anidado (nCRP), la distribución de Dirichlet y el muestreador de Gibbs, cuyos fundamentos se detallarán.

Finalmente, se validará el algoritmo implementado haciendo uso de dos bases de datos: NIPS y CORA.

Abstract

For some years now, machine learning techniques have evolved to become very important in everyday life. In the era of technology in which a large amount of data is generated constantly, it is necessary to use automatic techniques that allow the processing of this data and the extraction of information to draw conclusions. For example, the information that can be extracted from social media is of great interest for marketing campaigns, facilitating access to a large audience and focusing those campaigns on the sector that may be interested in them.

A field of action of machine learning is the data mining that is responsible for extracting and processing information from a corpus of documents: books, reviews, articles, tweets, ... and within data mining, are the techniques of topic modeling, which are in charge of obtaining the present themes is the texts. There are a large number of algorithms that try to solve this problem by attending various sources of information.

This project will focus on the investigation of the techniques used in the resolution of topic modeling problems present in the state of the art and in the choice and development of one of these, HLDA, which elaborates a hierarchical tree of topics where each document will follow one of the branches of this tree and the topics it deals with will correspond to each of the nodes of its branch. This algorithm will use complex models such as the nested Chinese restaurant process (nCRP), the Dirichlet distribution and the Gibbs sampler, whose foundations will be detailed.

Finally, the implemented algorithm will be validated using two databases: NIPS and CORA.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Notación

$\leq$	Menor o igual
$\geq$	Mayor o igual

1 OBJETIVO Y MOTIVACIÓN

El verdadero progreso es el que pone la tecnología al alcance de todos

-Henry Ford-

A lo largo de los últimos años, la cantidad de datos disponibles se ha vuelto tan grande y de tal variedad que el ser humano puede extraer gran cantidad de información. Nos encontramos en la era del Big Data. Ante esta inundación de información, es necesario desarrollar técnicas que permitan el manejo de los datos. La información que se puede extraer es muy variada: estados de ánimo, intereses de las personas, incluso predecir el comportamiento. Estos métodos se engloban en lo que se conoce como machine learning (ML) o aprendizaje máquina. En particular, se define el ML (Murphy, 2012) como un conjunto de técnicas o métodos que permiten extraer patrones en los datos y usar otros patrones para realizar predicciones u otras decisiones que poseen cierta incertidumbre. En la actualidad, se invierte un gran esfuerzo en alcanzar soluciones a estos problemas, empleando estas técnicas automáticas en las que el factor humano se ve ampliamente reducido. En concreto, este proyecto busca poder manejar la información contenida en textos de diversa índole. Por ejemplo, para manejar la gran cantidad de artículos disponibles en cualquier plataforma, es necesario un sistema que clasifique estos textos y que, el usuario, haciendo una búsqueda pueda encontrar la información deseada. Este proyecto surge como un punto de partida en la investigación de técnicas de aprendizaje máquina para resolver el problema de clasificación de textos en función de su contenido (topic modeling).

Los problemas que actualmente intentan resolver los algoritmos de ML se pueden dividir en tres grandes grupos:

- Aprendizaje supervisado. En este caso, se busca obtener un dato de salida, llamada *etiqueta*, en función de los datos de entrada que se denominan *características*. Se dispone de un conjunto de datos de entrenamiento en los que las etiquetas son conocidas para poder llevar a cabo la primera fase del algoritmo, el aprendizaje. Las características pueden ser de distintos origen: edad, género de una persona, una imagen, textos, etcétera. Por otro lado, las etiquetas también pueden ser de diferentes clases: valores categóricos o no categóricos.
- Aprendizaje no supervisado. En estos tipos de problemas se dispone de un conjuntos de datos de entrada a los que se les quiere extraer los patrones que siguen para poder así obtener información interesante. A diferencia del caso anterior, no se dispone de etiquetas definidas. Estos problemas no están tan bien definidos como los anteriores debido a que no se conoce a priori qué patrón se busca, por tanto, la dificultad se ve incrementada.
- Aprendizaje con refuerzo. Es menos conocido que los anteriores y se basa en la idea de llevar a cabo el aprendizaje usando retroalimentación, feedback (Wiering, 2012). Se emplea en campos como la teoría de juegos, teoría de control, etcétera.

En un segundo nivel, más específico, se pueden diferenciar tres tipos de problemas de ML:

- Clasificación. Es un tipo de problema de aprendizaje supervisado que se caracteriza por tener datos de salida categóricos, es decir, datos que toman valores reales de un conjunto previamente definido. Su objetivo es obtener un dato de salida, y, en función de las características de entrada, $\boldsymbol{x}$. Se pueden diferenciar dos tipos de clasificadores: binarios, en los que la salida puede tomar dos valores, y multiclase, en los que la salida toma diversos valores. Con el clasificador se busca una función que relacione las entradas con una salida:

$$y = f(\boldsymbol{x})$$

  Esta función, sistema de clasificación, es desconocida y se desea encontrar aquella con la que se obtenga menor error de clasificación. Para ello, se emplea un conjunto de datos de entrenamiento cuyas etiquetas son conocidas y, posteriormente, se empleará esta función al resto de datos en los que la salida es desconocida. A esto se le llama *generalización*. Se debe obtener un clasificador que generalice correctamente a todos los datos de entrada.

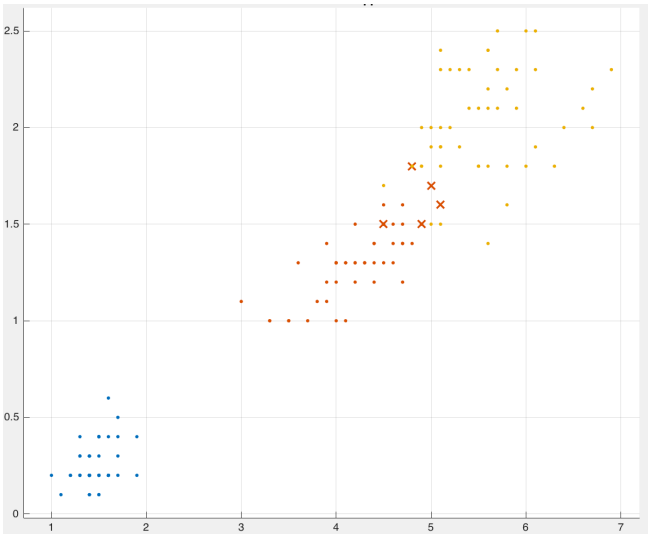

Figura 1. Ejemplo de clasificación usando support vector machine (SVM)

- Regresión. Este tipo de problemas se diferencian de los de clasificación en que la salida es una variable continua. Algunos ejemplos de regresión son: la predicción de la bolsa, predicción de localización o predicción de la temperatura. Se pueden encontrar diversas clases:
  a. Regresión lineal, que realiza la predicción empleando una función lineal.
  b. Regresiones no lineales, que pueden usar funciones cuadráticas, polinómicas, radiales, …

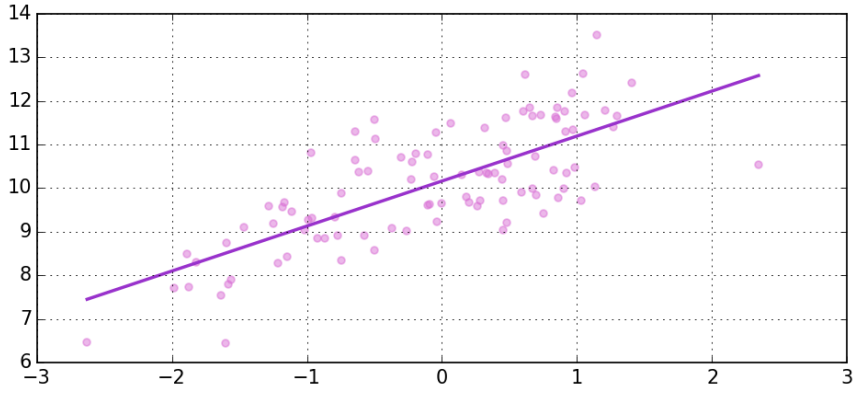

Figura 2. Ejemplo de Regresión

- Clustering. Es un problema no supervisado en la que los datos no están etiquetados y que busca poner los datos de entrada en diversos conjuntos. El número de conjuntos o clusters, K, se encuentra definido por el usuario. En la actualidad, hay diversas técnicas que buscan obtener el número de

clusters de forma automática, haciendo que la intervención humana sea mínima. Hay diferentes modelos en función de cómo se deseen agrupar los datos: k-means, basado en centroides y la distancia de los datos a éstos; algoritmo expectation-maximization (EM) que emplea distribuciones estadísticas para definir los clusters.

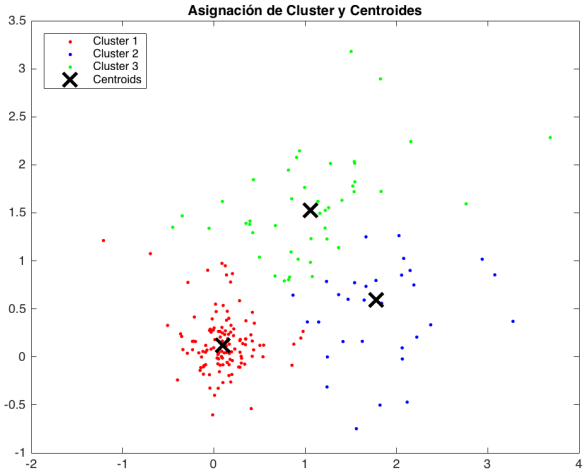

Figura 3. Ejemplo de clustering usando el algoritmo de K-means

Otra clasificación de los métodos de ML se basa en el conocimiento de los parámetros de éstos. De esta forma, se distinguen dos modelos (Alpaydin, 2014):

- Modelos paramétricos. En estos métodos, los parámetros del algoritmo están definidos en un conjunto finito. Por tanto, se puede ver que la intervención humana es necesaria. Tienen la ventaja de ser más rápidos. Sin embargo, cuentan con un mayor error por el factor humano.
- Modelos no paramétricos. En estos métodos, no se definen previamente los parámetros, ofreciendo una mayor flexibilidad de los algoritmos. Sin embargo, estos modelos se hacen intratables para conjuntos de datos (*dataset*) muy extensos. Un ejemplo de modelo no paramétrico es el estimador K-NN (K- nearest neighbor).

Este proyecto, como se ha mencionado anteriormente, tiene como objetivo clasificar unos documentos en función de su contenido. Es un problema en el que no se conoce previamente el tema del texto, sino que el usuario únicamente posee los textos como entrada para el algoritmo. Por tanto, con este proyecto se busca emplear técnicas de clustering para clasificar los documentos. Existen diversos métodos que pueden emplearse para extraer las palabras más significativas de los textos y diferenciarlos en función de éstas. *Latent Dirichlet Allocation*, *Named-Entity Recognition* o *Word2vec* son algunos de los algoritmos empleados en los problemas de Topic Modeling.

En el siguiente apartado se realizará una introducción a los problemas de aprendizaje automáticos definidos como Topic Modeling.

1.1. Introducción a Topic Modeling

En este apartado se tratará el problema de ML relacionado con el procesado de textos, en concreto, se introducirán algoritmos destinados a la extracción de información de dichos textos. Debido a la gran cantidad de datos y de contenido muy diverso de la que se dispone actualmente, el problema de minería de datos ha ido cobrando importancia en los últimos años. Hay una gran necesidad de diseñar algoritmos que procesen estos datos y extraigan información de forma eficiente (Aggarwal & Zhai, 2012). Estos algoritmos tienen como objetivo buscar patrones en los datos de forma dinámica y escalable. Entre las fuentes de información disponibles se encuentran: artículos científicos, periodísticos, correos electrónicos, publicaciones en redes sociales, … (Murphy, 2012)).

Los principales objetivos de la minería de datos son:

- Analizar los documentos en busca de patrones, tendencias, etcétera.
- Profundizar en los datos que presentan los textos para ayudar a los usuarios a extraer y procesar la información para facilitar toma de decisiones.

Para poder diseñar los algoritmos de manera eficiente es necesario conocer las características de los datos de entrada, en este caso, los textos. Estas características son la gran dimensión y la variedad del contenido. Por ejemplo, un conjunto de documentos, denominado *corpus*, puede contener sobre 100.000 palabras diferentes, mientras que uno de los documentos que lo componen contendría únicamente del orden de cientos de estas palabras. Estas características de los textos hacen necesaria la aplicación de técnicas que reduzcan la dimensión para hacer más manejables los datos.

Además, otro de los aspectos a tener en cuenta en el diseño de los algoritmos es el origen de los datos. A veces será necesario aplicar técnicas de lenguaje natural, obtener información semántica o usar lenguaje específico, por ejemplo, en textos biomédicos.

Los algoritmos de minería de datos se pueden organizar en los siguientes tipos (Aggarwal & Zhai, 2012):

- Extracción de información. Es unos de los principales problemas de la minería de datos que se emplea como punto de partida para otros algoritmos. Por ejemplo, la extracción de la relación semántica de las palabras de un texto puede servir para llegar hasta la información más "enterrada" en el texto, es decir, llegar hasta un nivel más profundo de conocimiento.
- Resumen del texto. Estos algoritmos buscan extraer la información principal de un texto de forma que se pueda crear una breve descripción del mismo u organizar los documentos en función de su tema principal.
- Métodos de aprendizaje no supervisado. Se pueden distinguir dos métodos: *clustering* y *topic modeling*. El primero de ellos buscar segmentar el corpus de documentos en distintas particiones, *clusters*. En el segundo, se busca emplear modelos probabilísticos para hacer una segmentación suave, *soft clustering*, en la que cada documento tendrá una probabilidad de pertenecer a un cluster. Cada tema, *topic*, se considerará una distribución de probabilidad de las palabras donde aquéllas que sean más representativas tendrán una mayor probabilidad. Se debe tener en cuenta que cada documento puede pertenecer a diversos topics. Otro caso es el *hard clustering*, en el que cada documento únicamente tiene un topic asociado.
- Reducción de dimensión. Es un método que busca la información más importante contenida en los documentos para poder comprimirlos. Una de las técnicas más conocidas es *latent semantic indexing* (LSI).
- Métodos de aprendizaje supervisado. Usa clasificadores o regresores para realizar predicciones.
- Aprendizaje de transferencia de textos. Parte del hecho de que mucha de la información se encuentra en un único idioma. Lo que se busca con estos métodos es transferir el conocimiento de un dominio a otro. Esto no sólo se puede aplicar a los idiomas sino también a la transferencia de datos entre diversos medios. Por ejemplo, en algunos artículos la información que posee el texto se complementa con algunas imágenes o vídeos, por tanto, si se pudiera transferir los datos de todos los medios, la información extraída sería mucho más completa. A esto se le conoce como *cross-media transfer*.
- Uso de minería de datos en vivo. Con la llegada de las redes sociales la información publicada se incrementa continuamente. Este tipo de datos supone un reto para la minería de datos debido a que, en ocasiones, resulta muy complicado guardar dichos datos offline y debe procesar la información de forma continua, a medida que se van generando los datos.
- Idiomas en la minería de datos. Como se ha mencionado anteriormente, algunos tipos de documentos únicamente están disponibles en un idioma y resulta interesante traducirlos a otros. Sin embargo, también puede ocurrir que varios textos sobre el mismo tema estén disponibles en distintos idiomas y, en el caso de que el usuario realice una búsqueda, sería interesante que encontrara todos los documentos sobre dicho tema.
- Minería de datos en redes sociales. Uno de los retos a los que se enfrenta la minería de datos es el uso del lenguaje en los textos de las redes sociales. Han cobrado mucha importancia debido a que la información contenida en las publicaciones en redes sociales resulta muy útil a nivel comercial y para influir a los usuarios. Estos textos se caracterizan por un contenido dinámico que, en muchas

ocasiones, contiene un lenguaje pobre y no estándar. Los algoritmos deberán tener en cuenta este uso del lenguaje para extraer información de forma óptima.

Este proyecto se centra en el estudio de técnicas de topic modeling (TM) que hace referencia a un modelo estadístico que tiene dos objetivos fundamentales: descubrir la estructura semántica subyacente en documentos, denominados *topics*, y buscar similitudes entre textos. Los modelos tienen en cuenta que un documento puede contener múltiples temas (Blei & Lafferty, Topic Models, 2009). Descubriendo estas estructuras es posible encontrar relaciones entre documentos y así poder estructurar colecciones de textos.

En los últimos años, las técnicas de topic modeling han ganado popularidad debido al incremento de los textos digitalizados disponibles. Estos métodos permiten el acceso a un gran conjunto de textos, *corpus*. Además, estos algoritmos no sólo pueden aplicarse a documentos, sino que también pueden analizar canciones (Nishma Laitonjam, 2015) o incluso imágenes.

Los modelos se pueden clasificar atendiendo a tres criterios:

- Basado en ordenación de palabras y representación de documentos. Podemos encontrar dos soluciones. Aquélla que no tiene en cuenta el orden de las palabras, sino que se centra en la estructura semántica del texto. Esto se conoce como *bag of words*. La otra solución sí considera el orden de las palabras, consiguiendo una mayor información sobre el documento.
- Basado en la consideración de información externa. Aquellas soluciones que no tienen en cuenta la información externa son modelos más simples y con buenos resultados. Por otro lado, las soluciones que sí lo consideran obtienen resultados más precisos.
- Basado en el etiquetado de los datos. La mayoría de los algoritmos se engloban en los modelos no supervisados, en los que los datos no están etiquetados. Sin embargo, hay algunos modelos que usan datos etiquetados para llevar a cabo una clasificación.

Un posible campo de aplicación de estos algoritmos puede ser la ordenación de un repositorio de papers de diversas categorías. Este problema tendría como entradas el corpus de documentos que forman la colección del repositorio y como salidas extraería cada uno de los temas subyacentes en los textos, permitiendo así una organización temática de los textos y facilitar al usuario la realización de una búsqueda en dicho repositorio. Se ha de tener en cuenta que, dependiendo del campo de aplicación, se debe emplear un conjunto de documentos más específico o más genérico. Por ejemplo, para organizar estudios del campo de la medicina, previamente se deberá hacer una selección de documentos que pertenezcan a este campo para, después, aplicando algún algoritmo de TM, hacer una separación más selectiva.

Respecto al corpus de documentos, en la actualidad hay un gran número de bases de datos disponibles para la utilización en el desarrollo de algoritmos de TM, por ejemplo, UCI es un repositorio especializado en conjuntos de datos que pueden emplearse para distintos problemas de ML, en concreto, *Bag of Words Data Set*, se compone de varios corpus de diferentes categorías: e-mails, papers de NIPS, entradas de blogs, artículos del New York Times y abstracts de PubMed (Center of Machine Learning and Intelligent Systems, University of California, s.f.). De este repositorio se extraerán uno de los corpus empleados en la validación del algoritmo propuesto en este proyecto.

Existen multitud de modelos que se centran en los problemas de TM. Sin embargo, la mayoría comparten una estructura funcional. Crean un corpus de documentos, de los cuales extraen todas las palabras que lo componen, vocabulario. Es a este último al que le aplican técnicas probabilísticas, dependientes del modelo empleado, para generar cada uno de los topics. Finalmente, se asociará cada documento, visto como conjunto finito de palabras, a los topics extraídos.

La siguiente figura muestra la clasificación de los algoritmos de topic modeling atendiendo a los criterios previamente comentados:

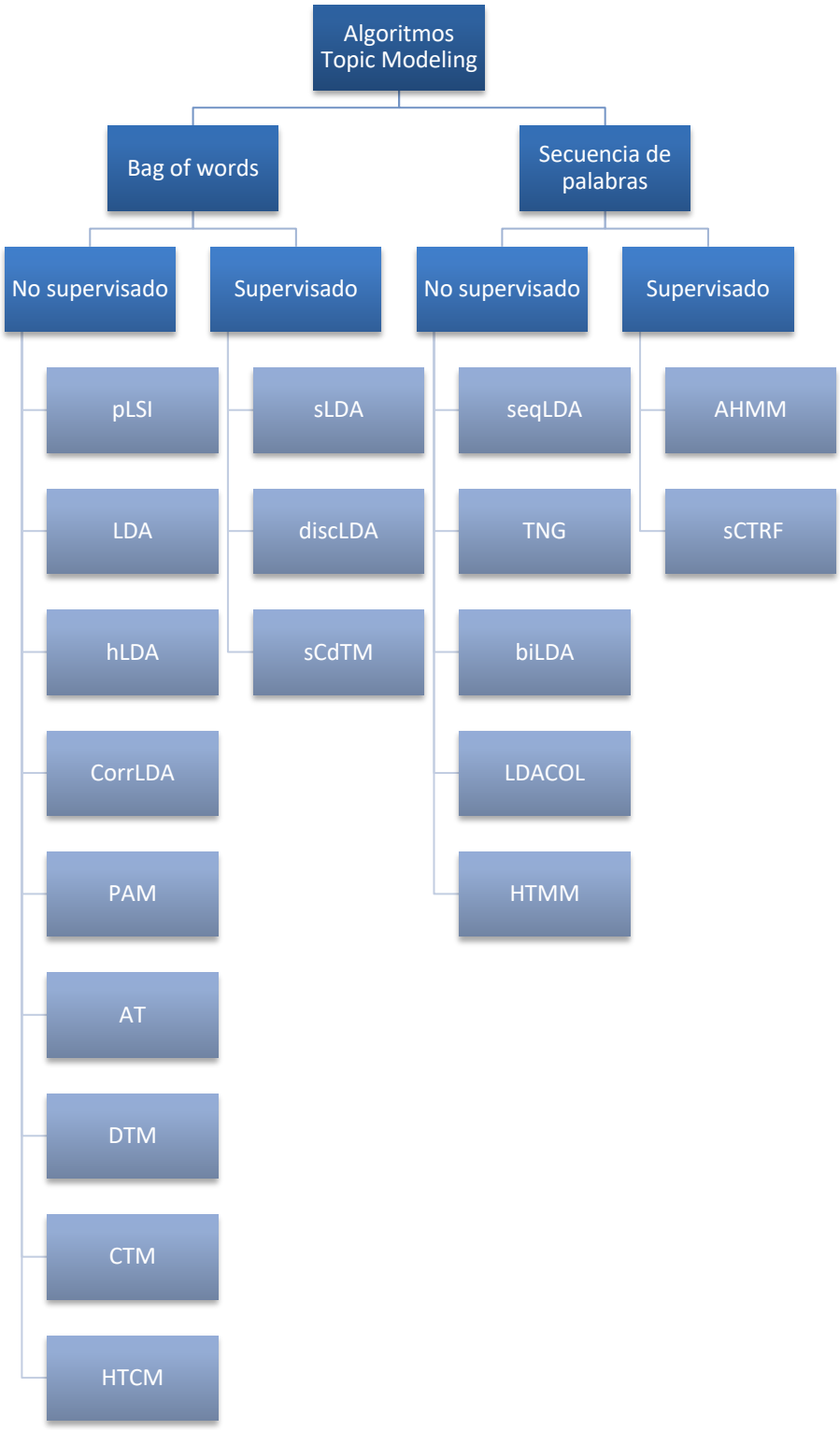

Figura 4. Clasificación de los algoritmos de Topic Modeling (V. Jelisavčić, 2012)

La primera técnica de TM fue propuesta por Blei et al. (D. M. Blei, 2003) y se conoce como Latent Dirichlet Allocation (LDA).

En el siguiente capítulo se llevará a cabo un estudio del estado del arte en el que se detallarán cada uno de los algoritmos de Topic Modeling.

2 ESTADO DEL ARTE

La curiosidad es una de las más permanentes y seguras características de una vigorosa inteligencia

-Samuel Johnson-

En este capítulo se introducirán los distintos algoritmos que buscan resolver el problema de Topic Modeling. Estos modelos hacen referencia a los métodos de machine learning que se dedican a encontrar la estructura intrínseca de un conjunto de documentos. Entre las características generales de estos algoritmos se hallan la consideración de que cada documento se compone de varios temas o topics y que cada tema se supone un conjunto de palabras que lo representa.

Tal y como se introdujo en el capítulo anterior, hay diversos algoritmos que se pueden clasificar atendiendo a distintos criterios. En los siguientes apartados se detallan cada unos de esos métodos, ordenándolos según su fecha de publicación.

2.1 Probabilistic Latent Semantic Analysis (pLSA)

Este modelo fue introducido en 1999 y es una técnica del área de procesado del lenguaje natural NLP, Natural Language Processing) que representa el contenido de los textos en forma de vector y calcula la similitud de estos textos en función de las palabras que contenga (Hofmann, 1999). La finalidad de este algoritmo, como su nombre indica, es extraer información de las relaciones semánticas de los documentos.

Es un modelo generativo que transforma estos vectores en una representación de menor dimensión en la que todo el corpus queda reflejado. Ese espacio se denomina, *latent semantic space*. Se supone un conjunto de documentos $\mathcal{D} = \{d_1, \dots, d_N\}$ que contiene palabras del vocabulario $\mathcal{W} = \{w_1, \dots, w_M\}$. La representación de los datos se puede hacer mediante una matriz $\mathcal{N}$de tamaño NxM en la que cada elemento $n(d_i, w_j)$ es el número de ocurrencias de la palabra w_j en el documento d_i. Para ello hace uso de la Descomposición en Valores Singulares (SVD, Singular Value Decomposition) que es una técnica que emplea una matriz para obtener $\mathcal{N}$.Esta representación se emplea en los algoritmos de la clase bag-of-words en los que la relación entre palabras no se tiene en consideración.

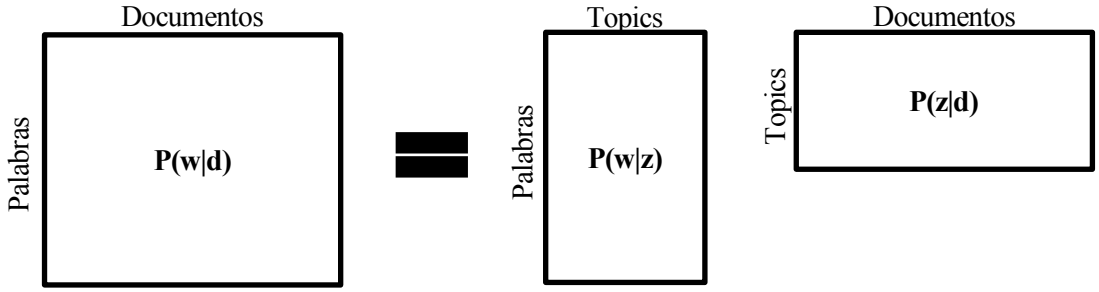

Figura 5. Representación de los datos en pLSA (V. Jelisavčić, 2012)

El algoritmo pLSA parte de un modelo de variable latente de la coocurrencia de los datos. Asocia una variable no observada, $z \in \mathcal{Z} = \{z_1, \dots, z_K\}$ a una observación. Se define un modelo de probabilidad conjunta:

$$P(d, w) = P(d)P(w|d)$$

en el que la probabilidad condicional viene dada por:

$$P(w|d) = \sum_{z \in \mathcal{Z}} P(w|z)P(z|d)$$

Este modelo considera que las variables d y w son estadísticamente independientes condicionadas al estado de la variable latente z. Esto se conoce como modelo de parametrización asimétrica y su representación se muestra en la siguiente figura.

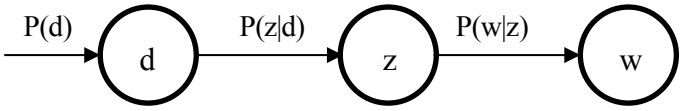

Figura 6. Modelo asimétrico

En esta parametrización cada clase se elige condicionado al documento y cada palabra se genera dependiendo de la clase (Alghamdi & Alfalqi, 2015).

Debido a que los valores de z son mucho menores que el número de palabras/documentos, esta variable puede actuar como cuello de botella y conviene emplear otro modelo equivalente de parametrización simétrica:

$$P(d, w) = \sum_{z \in \mathcal{Z}} P(z)P(d|z)P(w|z)$$

Este segundo modelo se caracteriza por la obtención tanto del documento como de la palabra a partir de la clase. La representación de este modelo es la siguiente:

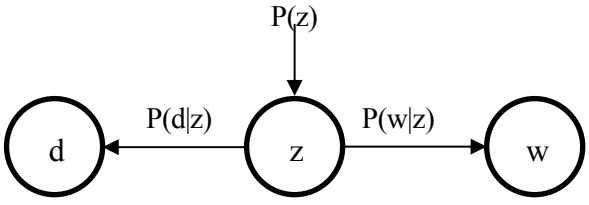

Figura 7. Modelo simétrico

Para la estimar la máxima verosimilitud del modelo pLSA se emplea el algoritmo Expectation Maximization

(EM), un método iterativo se caracteriza por emplear la divergencia de Kullback-Leibler como función objetivo y por alternar dos pasos:

- Expectation, donde se calcula las probabilidades a posteriori de las variables latentes. Para ello se emplea la siguiente expresión:

$$P(z|d,w) = \frac{P(z)P(d|z)P(w|z)}{\sum_{z' \in \mathcal{Z}} P(z')P(d|z')P(w|z')}$$

- Maximization, donde se actualizan los parámetros atendiendo a las siguientes expresiones:

$$P(w|z) \propto \sum_{d \in \mathcal{D}} n(d,w)P(z|d,w)$$

Finalmente, con el fin de conseguir una mayor generalización, se emplea una variación del algoritmo EM, tempered EM, que añade un parámetro de control a la estimación para evitar el overfitting y emplea otra función objetivo, energía libre de Helmholtz:

$$\mathcal{F}_\beta = -\beta \sum_{d,w} n(d,w) \sum_z \widetilde{P}(z;d,w) log P(d,w|z)P(z) + \sum_{d,w} n(d,w) \sum_z \widetilde{P}(z;d,w) \log \widetilde{P}(z;d,w)$$

El modelo pLSA se caracteriza además por tener en cuenta las palabras polisémicas, consiguiendo en esas situaciones mejores resultados que las técnicas LSA. Sin embargo, presenta algunos inconvenientes, por ejemplo:

- El número de parámetros a estimar depende del tamaño del corpus. Esto puede causar problemas de sobreestimación al incrementar el número de documentos.
- Puede no considerarse un modelo generativo debido a que únicamente tiene en cuenta el vocabulario presente en el set de entrenamiento para generar los documentos.

Estos problemas se resuelven con el modelo presentado en (D. M. Blei, 2003), Latent Dirichlet Allocation (LDA).

2.2 Latent Dirichlet Allocation (LDA)

Es un modelo probabilístico generativo, no supervisado y no parametrizado que mejora pLSA, introduciendo priors en las distribuciones que relaciona cada documento con un topic. Fue publicado en 2003 por D. Blei, A. Ng y M. Jordan y se presenta como el primer algoritmo de Topic Modeling.

Resuelve el problema del pLSA del incremento de parámetros a estimar, empleando la distribución de Dirichlet que considera cada parámetro como una variable aleatoria (V. Jelisavčić, 2012). Con esta distribución también se consigue que el modelo sea plenamente considerado generativo. Sin embargo, estas mejoras hacen que el algoritmo sea más complejo, haciéndolo incluso intratable a menos que se empleen aproximaciones con algoritmos MCMC (Markov Chain Monte Carlo), por ejemplo, Gibbs Sampling.

Al igual que pLSI, considera que cada documento es una distribución de topics y que cada uno de estos topic es una distribución de probabilidad de las palabras de un documento. Considera a los documentos como vectores en los que cada posición se corresponde con el número de veces que está presenten una palabra en el documento. Se considera un modelo generativo debido a que los topics se especifican antes de generar los datos.

El modelo LDA se caracteriza por realizar las siguientes suposiciones:

- Los topics son distribuciones de un vocabulario fijo. Se asumen *K* topics en la colección.
- Cada documento contiene estos topics en diferente proporción. El algoritmo quiere encontrar cada topic de los documentos del corpus.

LDA se basa en un modelo de variables ocultas que se basan en la interacción de los datos observados con variables aleatorias ocultas. En este caso, los datos observados son cada uno de los textos y las variables ocultas son los topics de cada documento. Dada la distribución del posterior de las variables ocultas, se determina la descomposición de tópicos del corpus.

Se suponen K topics; V palabras en el vocabulario; $\vec{\alpha}$, un vector de tamaño K que se emplea como parámetro de la distribución de Dirichlet que define la proporción de los topics de cada documento y η, un escalar empleado en la distribución de Dirichlet que define la distribución de cada topic en función de las palabras (Blei & Lafferty, Topic Models, 2009).

El algoritmo LDA se puede resumir en los siguientes pasos:

1. Para cada topic:
   a. Definir una distribución de las palabras. $\vec{\beta}_k \sim Dir_V(\eta)$
2. Para cada documento:
   a. Definir un vector de proporciones de topic. $\vec{\theta}_d \sim Dir(\vec{\alpha})$
   b. Para cada palabra:
      i. Asignar un topic. Para ello se sigue una distribución multinominal.

         $Z_{d,n} \sim Mult(\vec{\theta}_d), Z_{d,n} \in \{1, \dots, K\}$

      ii. Muestrear una palabra del vocabulario. Se escoge siguiendo una distribución multinomial.

         $W_{d,n} \sim Mult\left(\vec{\beta}_{Z_{d,n}}\right), W_{d,n} \in \{1, \dots, V\}$

La siguiente figura, muestra un gráfico del modelo LDA, donde cada nodo representa una variable aleatoria y las flechas, las dependencias entre variables.

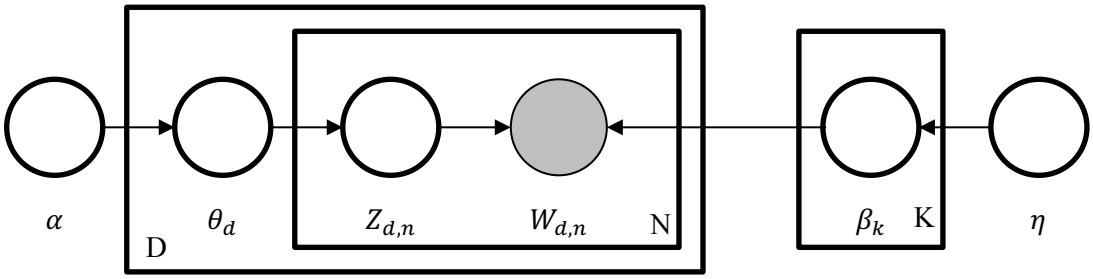

Figura 8. Representación gráfica del modelo LDA

$\vec{\beta}_{1:K}$ representa la estructura de topics; $\vec{\theta}_{1:D}$, las proporciones de cada topic por documento y $z_{1:D,1:N}$, la asignación de un topic a cada palabra.

Como se ha mencionado anteriormente, el modelo LDA hace uso de la distribución de Dirichlet que pertenece a la familia de las distribuciones exponenciales y se caracteriza porque sus valores suman uno. El algoritmo LDA tiene dos variables Dirichlet, la proporción de los topics, $\vec{\theta}_d$, y la distribución de estos topics por el vocabulario, $\vec{\beta}_k$.

El modelo LDA proporciona la distribución conjunta de las variables aleatorias ocultas y observadas. Para hallar la descomposición de topics se calcula el posterior de las variables ocultas dados los documentos.

$$p(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K} | w_{1:D,1:N}, \alpha, \eta) = \frac{p(\vec{\theta}_{1:D}, \vec{z}_{1:D}, \vec{\beta}_{1:K} | \vec{w}_{1:D}, \alpha, \eta)}{\int_{\vec{\beta}_{1:K}} \int_{\vec{\theta}_{1:D}} p(\vec{\theta}_{1:D}, \vec{z}_{1:D}, \vec{\beta}_{1:K} | \vec{w}_{1:D}, \alpha, \eta)}$$

Esta expresión es intratable debido a la integral del denominador. Por tanto, es necesario aplicar métodos de aproximación. Cada una de ellas posee ventajas e inconvenientes, pero es muy importante elegir aquella en la que exista un compromiso entre la velocidad, la complejidad, la precisión y la simplicidad conceptual. Existen diversas técnicas:

- Inferencia variacional de campo medio (D. M. Blei, 2003). Este método busca simplificar la expresión del posterior empleando una parametrización que se acerque al verdadero valor. Parte de la idea de que el motivo de que la expresión sea intratable es la dependencia existente entre las variables. Por tanto, se harán uso de unos parámetros con los que se conseguirá la independencia de dichas variables.

Se asignará un parámetro a cada una de las variables. Finalmente, el cálculo del posterior se realiza con la siguiente expresión:

$$q(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K}) = \prod_{k=1}^{K} q(\vec{\beta}_k|\vec{\lambda}_k) \prod_{d=1}^{D} \left(q(\vec{\theta}_d|\vec{\gamma}_d) \prod_{n=1}^{N} q(z_{d,n}|\vec{\phi}_{d,n}) \right)$$

donde cada variable oculta se describe por una distribución: $\vec{\beta}_{1:K}$ son los topics cuyos valores vienen dados por la distribución de Dirichlet de parámetro $\vec{\lambda}_k$; la proporción de los topics $\vec{\theta}_{1:D}$ viene dada por la distribución de Dirichlet de parámetro $\vec{\gamma}_d$; y la asignación del topic viene descrita por una distribución multinomial de parámetro $\vec{\phi}_{d,n}$.

Una vez calculado el posterior, la función objetivo a minimizar viene dada por la distancia Kullback-Leibler (KL):

$$arg \min_{\vec{\gamma}_{1:D}, \vec{\lambda}_{1:K}, \vec{\phi}_{1:D,1:N}} KL(q(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K}) || p(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K}|w_{1:D,1:N}))$$

Esta función objetivo no es tratable y debe reemplazarse por otra que hace uso de los parámetros previamente introducidos:

$$\mathcal{L} = \sum_{k=1}^{K} E[\log p(\vec{\beta}_k|\eta)] + \sum_{d=1}^{D} E[\log p(\vec{\theta}_d|\vec{\alpha})] + \sum_{d=1}^{D} \sum_{n=1}^{N} E[\log p(Z_{d,n}|\vec{\theta}_d)] + \sum_{d=1}^{D} \sum_{n=1}^{N} E[\log p(w_{d,n}|Z_{d,n}, \vec{\beta}_{1:K})] + H(q)$$

Donde H hace referencia a la entropía.

- Muestreo de Gibbs (Griffiths & Steyvers, 2004). Método propuesto por Griffiths y Steyvers que toma muestras del posterior para aproximarlas con una distribución. Se considera uno de los métodos de Monte Carlo de cadenas de Markov (*MCMC, Markov Chain Monte Carlo*). Estos métodos construyen cadenas de markov con la distribución deseada a partir de una serie de muestras. El proceso de muestreo comienza tras la ejecución de un gran número de pasos para asegurar la estabilidad. Este método obtiene la distribución condicional de la variable latente z a partir de la expresión:

$$P(z_i = j|z_{-i}, w) \propto \frac{n_{-i,j}^{(w_i)} + \beta}{n_{-i,j}^{(\cdot)} + W\beta} \frac{n_{-i,j}^{(d_i)} + \alpha}{n_{-i}^{(d_i)} + K\alpha}$$

donde:

$n_{-i,j}^{(w_i)}$ es el número de veces que aparece la palabra w_i en el topic j

$n_{-i,j}^{(\cdot)}$ es el número total de palabras asignadas al topic j

$n_{-i,j}^{(d_i)}$ es el número de palabras del documento d_i asignadas al topic j

n^{d_i} es el número total de otras palabras en el documento d_i

Para realizar el muestreo se inicializa la variable z_i a un valor comprendido entre 1 y el número total de topics K y se va actualizando en cada iteración con la expresión. Tras numerosas iteraciones, cuando se alcanza una distribución estacionaria, se tomarán las muestras de z_i.

2.3 Hierarchical LDA (hLDA)

Este modelo es una extensión del método LDA propuesto por Blei en 2003 (Blei, Griffiths, Jordan, & Tenebaum, Hierarchical Topic Models and the Nested Chinese Restaurant Process, 2013) que organiza los topics en una estructura en árbol en lugar de la estructura plana usada en LDA. Este modelo se caracteriza por combinar el cálculo del prior con la verosimilitud basándose en la variación jerárquica del LDA. Dado un corpus en el que cada documento contiene unas palabras, se busca descubrir los topics presentes en cada documento, organizándolos jerárquicamente. Emplea una aproximación bayesiana no paramétrica que permite construir el árbol de topics conforme van llegando los datos, este modelo se denomina *nested Chinese Restaurant Process* (nCRP) y es una adaptación del algoritmo CRP para emplearlo con jerarquías. Cada nodo de la estructura representa una variable aleatoria que tiene asociada la distribución palabra-topic. Este algoritmo tiene en cuenta dos consideraciones: se debe definir la profundidad del árbol y cada documento queda asociado a una rama de dicho árbol, aunque idealmente pueda pertenecer a varias.

Considerando el conjunto de datos de entrada, corpus, formado por documentos que contienen palabras de cierto vocabulario, el modelo asume que estas palabras se han generado siguiendo un método de mezcla de palabras donde cada proporción, θ, es aleatoria. Cada una de estas palabras está asociada a uno o varios topics, z. El modelo parte de un árbol previamente definido de L niveles y define un proceso de tres pasos para generar los documentos:

- Primero, se elige un camino desde un nodo raíz a una hoja.
- Segundo, se define un vector de proporciones de topics, θ, siguiendo la distribución de Dirichlet.
- Tercero, se generan las palabras del documento mediante una mezcla de topics desde el nodo raíz al nodo hoja empleando el vector previamente calculado, θ.

El algoritmo CRP se emplea para generar la estructura del árbol de forma aleatoria, define el prior de las proporciones de cada topic.

Finalmente, el algoritmo hLDA puede estructurarse en:

1. Elegir el nodo raíz, c_1
2. Para cada nivel $l \in \{2, \dots, L\}$:
    a. Elegir el nodo, c_{l-1}, atendiendo a las expresiones del algoritmo de nCRP:

$$p(nodo\ ocupado\ i \mid palabras\ previas) = \frac{m_i}{\gamma + m - 1}$$

$$p(siguiente\ nodo\ desocupado \mid palabras\ previas) = \frac{\gamma}{\gamma + m - 1}$$

    b. Elegir nodo c_l a partir del anterior.
3. Calcular el vector de proporciones de topics, θ, empleando la distribución de Dirichlet.
4. Para cada palabra $n \in \{2, \dots, N\}$:
    a. Seleccionar un topic $z \in \{2, \dots, Z\}$ siguiendo la distribución multinomial.
    b. Seleccionar las palabras asociadas al topic c_z.

Este modelo se ilustra en la siguiente figura, donde el nodo T hace referencia al conjunto infinito de ramas de L niveles creados con el algoritmo nCRP. Dado un T, la variable $c_{m,l}$ hace referencia al topic asociado al nivel l y al camino m. En el caso de que llegue un nuevo documento que no está reflejado en el árbol, se empleará el algoritmo nCRP para asociarlo a su camino. Se debe tener en cuenta que este nuevo documento puede compartir camino con otro documento previamente asociado, o partir de uno completamente nuevo. Esto se traduce en que el posterior de un nuevo documento puede depender del posterior de los documentos previamente procesados.

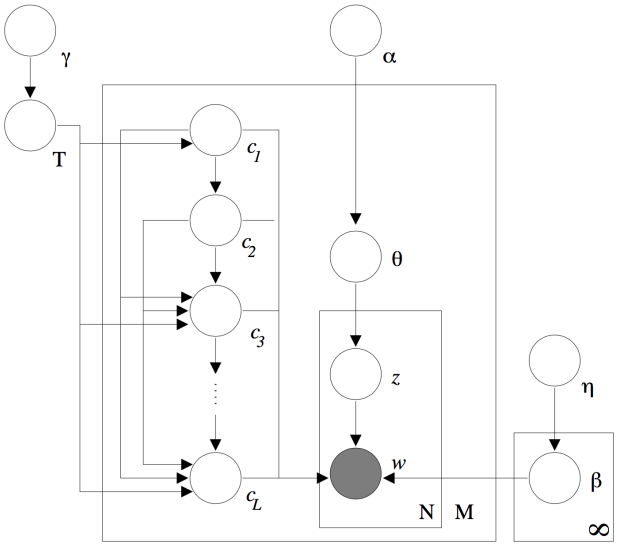

Figura 9. Modelo hLDA (Blei, Griffiths, Jordan, & Tenebaum, Hierarchical Topic Models and the Nested Chinese Restaurant Process, 2013)

Al igual que en el método LDA, el posterior requiere de una aproximación para poder ser calculable. En este caso, el método propuesto por Blei es el muestreo de Gibbs, detallado en el apartado anterior. Las variables necesarias son: $w_{m,n}$, corresponde a la n-ésima palabra del m-ésimo documento; $c_{m,l}$, el nodo correspondiente al l-ésimo topic en el m-ésimo documento; y $z_{m,n}$, que corresponde a la asignación de la n-ésima palabra del m-ésimo documento a uno de los L topics disponibles. Con el método de Gibbs se quiere muestrear las variables $c_{m,l}$ y $z_{m,n}$.

El proceso de muestreo se divide en dos partes:

1. Muestrear $z_{m,n}$ con el método descrito en LDA.
2. Dados los valores de las variables ocultas en LDA, muestrear $c_{m,l}$, empleando el prior de CRP. La distribución de los L topics asociados al documento m viene dado por:

$$p(\boldsymbol{c}_m|\boldsymbol{w},\boldsymbol{c}_{-m},\boldsymbol{z}) \propto p(\boldsymbol{w}_m|\boldsymbol{c},\boldsymbol{w}_{-m},\boldsymbol{z})p(\boldsymbol{c}_m|\boldsymbol{c}_{-m})$$

Para calcular la verosimilitud es necesario hacer la aproximación:

$$p(\boldsymbol{w}_m|\boldsymbol{c},\boldsymbol{w}_{-m},\boldsymbol{z}) = \prod_{l=1}^{L} \frac{\Gamma\left(n_{c_{m,l},-m}^{(\cdot)} + W\eta\right)}{\prod_w \Gamma\left(n_{c_{m,l},-m}^{(w)} + \eta\right)} \frac{\prod_w \Gamma\left(n_{c_{m,l},-m}^{(w)} + n_{c_{m,l},m}^{(w)} + \eta\right)}{\Gamma\left(n_{c_{m,l},-m}^{(\cdot)} + n_{c_{m,l},m}^{(\cdot)} + \eta\right)}$$

donde:

$n_{c_{m,l},-m}^{(w)}$ es el número de veces que aparece la palabra w en el topic $c_{m,l}$ en los documentos que no sean el actual.

W es el tamaño del vocabulario.

$\Gamma(\cdot)$ es la función gamma estándar.

2.4 Dynamical Topic Model (DTM)

Es un modelo publicado en 2006 (Blei & Lafferty, Dynamic Topic Models, 2006) y se presenta como una mejora del modelo LDA en el que se tiene en cuenta la evolución temporal de los topics en una colección de documentos. En los métodos propuestos anteriormente, los documentos se suponían independientes. Sin embargo, en numerosos textos la evolución temporal se considera un factor importante, por ejemplo, en los correos electrónicos, artículos periodísticos, etc. Por tanto, la fecha de los documentos que forman el corpus corresponde a un metadato empleado por el algoritmo.

El modelo DTM separa los datos por fecha, por ejemplo, agrupándolos por año. Se modelan los documentos

de cada grupo, teniendo en cuenta que los topics del grupo t evolucionan de los topics del grupo $t-1$. Este método no emplea las distribuciones de Dirichlet, como lo hacían los modelos anteriores, debido a que no resulta manejable. En su lugar, usa una distribución Gaussiana de media α para el cálculo de proporciones de los topics.

El proceso generativo para el grupo temporal t es:

1) Seleccionar los topics $\beta_t | \beta_{t-1} \sim \mathcal{N}(\beta_{t-1}, \sigma^2 I)$
2) Calcular las proporciones $\alpha_t | \alpha_{t-1} \sim \mathcal{N}(\alpha_{t-1}, \delta^2 I)$
3) Para cada documento:
    a. Definir $\eta \sim \mathcal{N}(\alpha_t, a^2 I)$
    b. Para cada palabra:
        i. Calcular el topic $Z \sim Mult(\pi(\eta))$
        ii. Obtener la palabra asociada al topic $W_{t,d,n} \sim Mult\left(\pi\left(\beta_{t,z}\right)\right)$

La representación gráfica de este modelo, como puede apreciarse en la figura, es más compleja. Cada columna corresponde a un grupo temporal y puede apreciarse cómo se encuentran relacionadas.

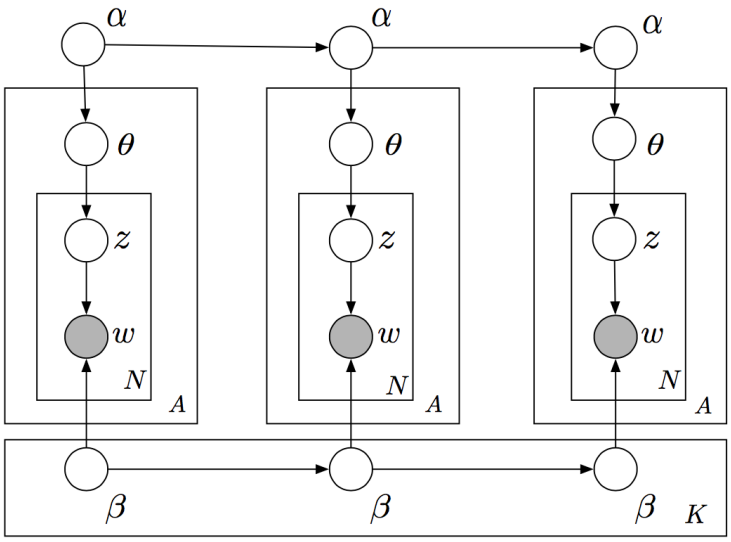

Figura 10. Representación gráfica del modelo DTM

En incremento de la complejidad de este modelo hace que el cálculo del posterior sea intratable. Además, los modelos de aproximaciones empleados en los métodos estáticos no son adecuados, sino que requiere de otros más complejos que buscan optimizar los parámetros de las variables latentes empleando la divergencia KL respecto al valor real del posterior, incorporando el factor temporal. Los posibles métodos de aproximación son:

- Variational Kalman Filtering. Este modelo no considera la existencia de topics, los cálculos son más simples $\hat{\beta}_t | \beta_t \sim \mathcal{N}(\beta_t, \hat{v}_t^2 I)$. Seguidamente, empleando el filtro de Kalman se obtiene la media y la varianza del posterior y, se repiten los pasos para cada grupo temporal.
- Variational Wavelet Regression. Emplea la descomposición de Wavelet para realizar las aproximaciones del posterior y el método de gradiente ascendente para obtener el óptimo.

El modelo DTM ofrece la ventaja de la introducción de la variable temporal en los datos de entrada para obtener nuevas relaciones entre documentos. Sin embargo, también cuenta con algunos inconvenientes como son: el número de topics fijo, la discretización del tiempo o la complejidad del algoritmo que crece rápidamente con la granularidad del tiempo que afecta a la carga computacional.

2.5 Correlated Topic Model (CTM)

Método propuesto por Blei en 2007 que mejora el modelo LDA introduciendo correlaciones entre topics (Blei

& Lafferty, A Correlated Topic Model of Science, 2007). En los modelos anteriores, cada uno de los topics se suponen independientes del resto. Con este modelo se quiere encontrar una estructura de documentos más compleja teniendo en cuenta las relaciones entre topics. Generalmente, este modelo se adapta mejor a los datos debido a la mayor información que se tiene en cuenta.

CTM hace uso de distribuciones normales para el cálculo de las proporciones de los topics, permitiendo ver la correlación entre éstos.

La notación que sigue este modelo es la siguiente:

- $w_{d,n}$ es la n-ésima palabra del documento d-ésimo.
- β_k es la distribución del topic k
- $z_{d,n}$ se refiere a la asignación de un topic a la n-ésima palabra del d-ésimo documento.
- θ_d es la proporción de cada topic del d-ésimo documento.

Este modelo sigue el proceso generativo:

1) Estimar la distribución de los topics $\theta = f(\eta) = \frac{\exp\{\eta\}}{\sum_i \exp\{\eta_i\}}$ donde η sigue una distribución multinomial.
2) Para cada palabra de cada documento
   a. Asignar un topic $z_{d,n}|\ \eta \sim Mult(\theta_d)$
   b. Asignar una palabra $w_{d,n}|\{z_{d,n}, \beta_{1:K}\} \sim Mult(\beta_{z_{d,n}})$

Este proceso se muestra en la siguiente imagen:

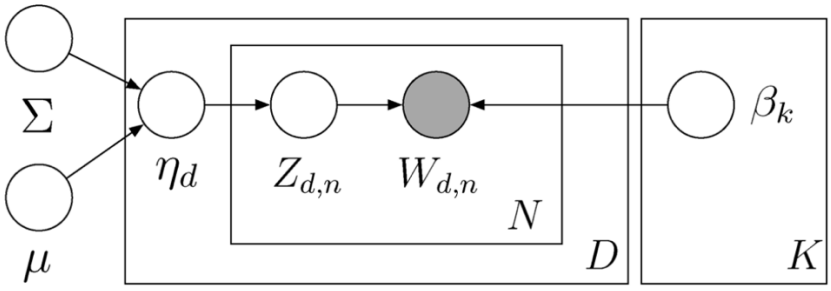

Figura 11. Representación gráfica CTM

Como puede observarse, este método es muy parecido al LDA salvo en la obtención de las proporciones de los topics. En este caso se emplea una distribución multinomial en lugar de la de Dirichlet.

Al igual que en LDA, se emplea el método de aproximación de inferencia variacional de campo medio para calcular los parámetros de forma que la divergencia KL entra la aproximación y el verdadero posterior sea mínima. Sin embargo, al emplear distribuciones normales, este proceso se complica.

2.6 Parachinko Allocation Model (PAM)

Método introducido en 2006 como alternativa a CTM (Li & McCallum, 2006). Al igual que este último, PAM tiene en cuenta las correlaciones entre topics para extraer la información oculta en los documentos. Sin embargo, para considerar las correlaciones se realiza una redefinición del concepto de topic. En este modelo se considera como una distribución tanto de las palabras como de otros topics. Para ello, hace uso del método conocido como *directed acyclic graph*, DAG, es un modelo en árbol, en el que cada nodo interno representa un topic que sigue una distribución de Dirichlet frente a sus hijos. Partiendo del nodo raíz, se muestrea uno de sus hijos siguiendo para ello la distribución multinomial y se continúa con el proceso hasta llegar a una hoja del árbol, una palabra.

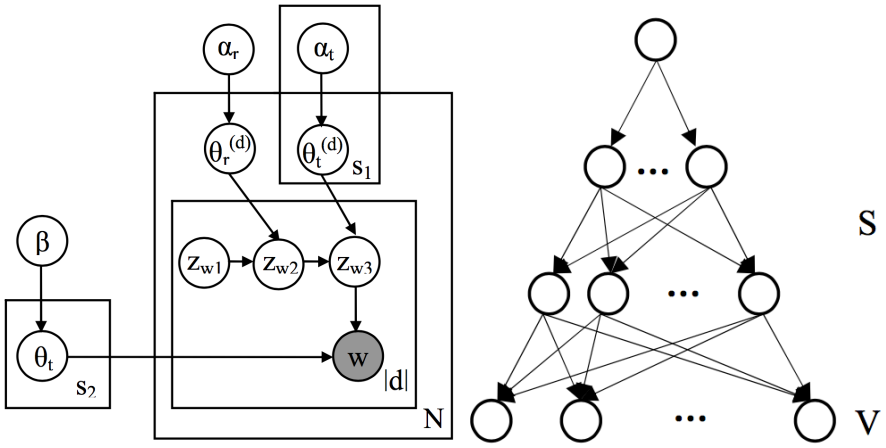

Figura 12. A la izquierda, representación gráfica del modelo PAM de 4 niveles. A la derecha, correlación en PAM de 4 niveles. (Li & McCallum, 2006)

Al igual que el resto de modelos, es necesario aplicar una aproximación para el cálculo del posterior. PAM propone hacer uso del muestreo de Gibbs, método previamente presentado.

PAM posee la ventaja de poder obtener correlaciones anidadas y no se restringe a usar la distribución normal. Además, soluciona el problema de escalado de CTM, donde únicamente se modela la correlación por pares.

2.7 Author Topic Model (ATM)

Se publica en 2004 como una extensión del modelo LDA (Rosen-Zvi, Griffiths, Steyvers, & Smyth, 2004) y, posteriormente, es revisado en 2010.

Se fundamenta en los mismos principios que los modelos anteriores, pero añade el uso de metadatos para extraer la información del autor del texto. Cada palabra de un documento se asociará a un topic y a un autor. Este modelo ve a cada autor como una distribución de topics y, al igual que en LDA, cada topic es una distribución de palabras.

Con estos modelos se puede extraer más información de los documentos, por ejemplo, los temas sobre los que escribe determinado autor o qué autores escriben sobre un determinado tema.

Parte de la idea de que un grupo de autores, a_d, deciden escribir sobre determinado tema. Cada palabra de esos documentos se elige por el autor siguiendo una distribución y, de la misma manera que en LDA, se elige el topic siguiendo una distribución tanto de palabras como de autores.

En la siguiente figura se representa el modelo ATM.

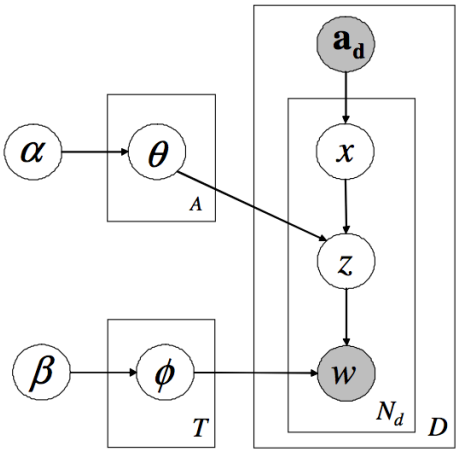

Figura 13. Modelo ATM

donde x es el autor correspondiente a una palabra, w, elegido del conjunto a_d. Los pesos de cada autor se emplean para seleccionar el topic z y cada palabra se genera siguiendo la distribución ϕ.

Como método de aproximación del posterior se emplea el muestreo de Gibbs debido a su simplicidad. En este caso, el número de variables latentes se incrementa. No sólo se considera la distribución de los topics, z, sino que también se ha de tener en cuenta la distribución de los autores, x.

2.8 Supervised LDA (sLDA)

A diferencia de los modelos anteriores, sLDA es un método supervisado que surge en 2007 como extensión del LDA que añade una variable observable, etiqueta, a cada documento (Blei & McAuliffe, Supervised Topic Models, 2007). Busca resolver los problemas de predicción, es decir, el objetivo del algoritmo es obtener aquellos topics que mejor se adapten a futuros documentos no etiquetados. Este modelo deberá modelar no sólo los documentos, como hacía LDA, sino también la respuesta de cada uno. Por tanto, los pasos a seguir son:

1) Modelar las proporciones de topics. $\vec{\theta}_d \sim Dir(\vec{\alpha})$
2) Para cada palabra:
    a. Asignar un topic. Para ello se sigue una distribución multinominal. $Z_{d,n} \sim Mult(\vec{\theta}_d)$
    b. Elegir una palabra del vocabulario. Se escoge siguiendo una distribución multinomial. $W_{d,n} \sim Mult(\vec{\beta}_{Z_{d,n}})$
3) Modelar la respuesta: $y \sim \mathcal{N}(\eta^T \bar{z}, \sigma^2)$ donde $\bar{z} = (\frac{1}{N}\sum_{n=1}^{N} z_n)$

Se puede observar que la única diferencia con el método LDA es la adición del modelado de la respuesta.

La siguiente imagen muestra la representación gráfica de este modelo.

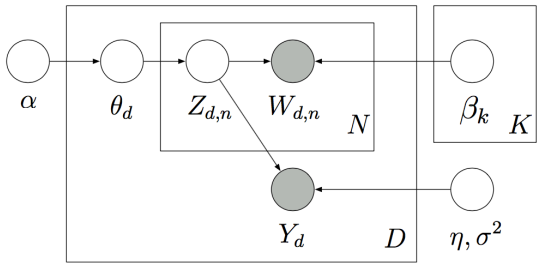

Figura 14. Representación del modelo sLDA

Por último, para realizar el cálculo del posterior se emplea el método de aproximación Expectation-Maximization (EM), al igual que en el modelo LDA. Con la diferencia de que es necesario añadir la variable respuesta.

2.9 Dirichlet Multinomial Regression (DMR)

Modelo publicado en 2008 como extensión del LDA cuya idea principal es la incorporación de metadatos de los documentos aleatoria sin una codificación adicional (Mimno & McCallum, 2008). Al igual que sLDA, es un método de aprendizaje supervisado.

La inclusión de los metadatos se realiza mediante un prior en las distribuciones que es una función de algunas características de los documentos, como son el autor, fechas o referencias.

Dado un documento, d, sea x_d un vector que contiene la característica que representa a los metadatos. Para cada topic, t, se dispone de un vector con la longitud del número de características, λ_t. Con estos parámetros definidos, el modelo generativo de DMR es:

1) Para cada topic t:

a. Estimar $\lambda_t \sim \mathcal{N}(0, \sigma^2 I)$
b. Estimar $\phi_t \sim Dir(\beta)$

2) Para cada documento d:
   a. Definir para cada topic $\alpha_{dt} = \exp(x_d^T \lambda_t)$
   b. Estimar la proporción de cada topic $\theta_d \sim Dir(\alpha_d)$
   c. Para cada palabra i
      i. Estimar el topic asociado: $z_i \sim Mult(\theta_d)$
      ii. Estimar las palabras asociadas al topic: $w_i \sim Mult(\phi_{z_i})$

Este modelo incluye tres hiperparámetros: σ^2, la varianza del prior; β, parámetro de las distribuciones topic-palabra; y T, el número de topics.

Posteriormente, se debe calcular la verosimilitud. Para ello, es necesario aplicar el método de muestreo EM en el que en un paso se muestrea la asignación de topics a partir del topic actual dadas unas características y, en el otro, se optimiza λ dada una asignación.

2.10 Topical N-Grams (TNG)

Modelo definido en 2007 que, al contrario de LDA que supone que cada documento es un simple conjunto de palabras (*bag of words*), tiene además en cuenta la consideración de la relación secuencial entre las palabras, siendo capaz de modelar frases complejas (Wang, McCallum, & Wei, 2007). Por tanto, este método no sólo descubre los topics de un documento sino también frases relacionadas con el topic.

El procedimiento de generación de palabras se puede ver en dos partes: primero, muestrea un topic asociado a dicha palabra y su estado como un bigram, una secuencia de dos elementos adyacentes; segundo, muestrea la palabra a partir de la distribución del bigram.

La siguiente imagen muestra la representación de este modelo. Este esta figura puede apreciarse la introducción del bigram en la variable σ, que sigue una distribución multinomial de parámetro δ, así como la variable ψ, que representa la distribución de las palabras teniendo en cuenta la palabra anterior.

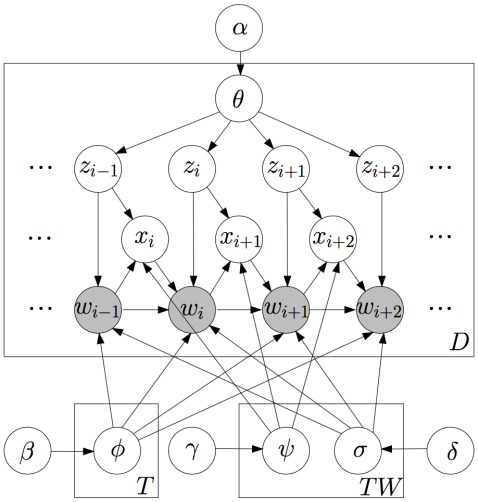

Figura 15. Representación gráfica del modelo TNG (Wang, McCallum, & Wei, 2007)

Para la aproximación del posterior se emplea, al igual que en otros métodos anteriormente explicados, el muestreo de Gibbs.

Como conclusión de TNG, es un modelo que emplea las relaciones semánticas permitiendo el modelado de conceptos o frases compuestas. Sin embargo, eso hace que los costes computacionales se vean incrementados considerablemente.

2.11 Aspect Hidden Markov Model (AHMM)

Modelo presentado en 2001 como combinación de HMM y aspect model (AM) para la aplicación en procesado de lenguaje natural (Blei & Moreno, 2001). Este modelo ve a cada documento como una colección de palabras mutuamente independientes, cada una de estas palabras ha sido generada siguiendo un variable oculta que refleja el topic. AHMM segmenta un nuevo documento tomando palabras o_t con una ventana de tamaño L y empleando el algoritmo de Viterbi para encontrar la secuencia de topics z más probable en esta ventana. Para la aproximación se emplea el método EM.

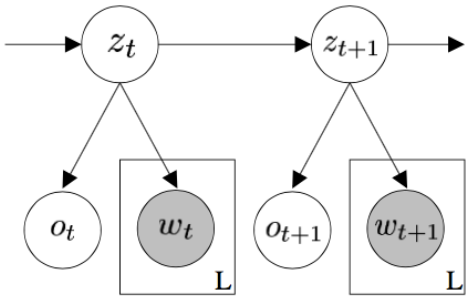

Figura 16. Representación de la segmentación AHMM (Blei & Moreno, 2001)

2.12 Named Entity Recognition (NER)

Named Entity Recognition es un problema de ML que busca categorizar palabras importantes en un texto. Entre sus diversas aplicaciones se encuentra la extracción de información de los documentos para una posterior clasificación (Mohit, 2014). Los algoritmos NER se introdujeron por primera vez en 1996 en la sexta conferencia sobre entendimiento del mensaje (MUC – Message Understanding Conference).

Antes de profundizar en los algoritmos es importante introducir el concepto *Named Entity*, que es una palabra o frase que identifica claramente una entidad (frase, párrafo, texto, etcétera) de otro conjunto con atributos parecidos. En MUC se definieron tres posibles etiquetas:

- ENAMEX, que incluye tres tipos: personas, organizaciones y lugares.
- TIMEX, que incluye fecha y tiempo.
- NUMEX, que hace referencia al dinero, porcentaje y cantidad.

Posteriormente, se introdujo otro estándar de etiquetado, denominado *BIO*, el cual utiliza las clases de la forma *B – X*, *I – X* y *O*, donde *X* es el nombre de una categoría (*PER* – persona, *ORG* – organización, *LOC* – localización), *B* corresponde al primer token (palabra) de la entidad *X*, *I* es un token de *X* que no es el primero y *O*, cuando no pertenece a ninguna categoría.

Otra alternativa es el formato *BILOU*, donde *L* identifica al último token y *U* a una entidad de una única palabra.

Originalmente, NER se empleaba para extraer personas, lugares y organizaciones. Sin embargo, con la entrada de otros dominios, es necesario introducir nuevos marcadores o clases.

Existen dos posibles ramas de algoritmos NER: basados en reglas y estadísticos. En los siguientes apartados se introducen cada uno de ellos.

2.12.1 NER basado en regla

Son los primeros sistemas de reconocimiento de entidades y emplean reglas manuales para detectar las entidades. La mayoría de estos sistemas presentan los siguientes componentes:

1) Reglas para la extracción de entidades
2) Diccionarios geográficos, gazeteers[1], para los diferentes tipos de las clases de entidades

[1] Gazeteers es un término empleado normalmente para referirse al léxico específico de un dominio.

3) Algoritmo de extracción de entidades que aplica las reglas definida

Las reglas pueden ser fijadas por el programador o aplicando técnicas de bootstrap a un conjunto de ejemplo.

Estos sistemas se caracterizan por ser relativamente precisos, pero con baja cobertura y consigue el mejor funcionamiento en dominios pequeños. Su funcionamiento depende en gran medida de lo comprensivas que sean las reglas y el léxico. Tiene el inconveniente de que la adición de reglas más complejas requiere un gran esfuerzo. Esto se soluciona con las técnicas estadísticas que son mucho más flexibles para incorporar mayor conocimiento lingüístico y más robustos.

2.12.2 NER estadístico

Buscan reducir la necesidad del factor humano para construir las reglas y los diccionarios. Estos sistemas constan de dos componentes principales:

1) Datos de entrenamiento etiquetados: corpus con las entidades ya etiquetadas.
2) Modelo estadístico: representación de los datos de entrenamiento. Este modelo emplea parámetros que mapean un evento del lenguaje en una probabilidad.

Es un modelo supervisado que puede emplearse para clasificación. Sin embargo, a diferencia de otros métodos, NER tiene en cuenta la secuencia de palabras y usa las sentencias para predecir.

Una sentencia se representa por un conjunto de tokens $t_1, \dots, t_N$ y el sistema debe encontrar las etiquetas de entidades que mejor las identifique $y_1, \dots, y_N$. Estadísticamente, se puede representar por:

$$S = \underset{y_1,\dots,y_N}{argmax}\, P(y_1, \dots, y_N | t_1, \dots, t_N)$$

Empleando el teorema de Bayes, esta expresión se reescribe:

$$S = \underset{y_1,\dots,y_N}{argmax}\, P(t_1, \dots, t_N | y_1, \dots, y_N)\, P(y_1, \dots, y_N)$$

Existen diversas técnicas de modelado. Una de las más conocidas es HMM (Hidden Markov Model) que se basa en dos conceptos:

- Un modelo probabilístico en el que las clases se representan por estados que son capaces de generar tokens.
- Supone que la generación de tokens sigue un proceso de Markov: la probabilidad de asignar una clase a un token depende de los tokens anteriores.

$$S = \underset{y_1,\dots,y_N}{argmax}\, P(t_1, \dots, t_N | y_1, \dots, y_N)\, P(y_1, \dots, y_N) = \prod_{i=1,\dots,N} P(t_i | y_i) P(y_i | y_{-i})$$

  Esta expresión representa a HMM de primer orden debido a que la generación únicamente depende la de palabra anterior.

La siguiente figura muestra la representación gráfica de HMM. Puede apreciarse que usa el formato BIO de etiquetado.

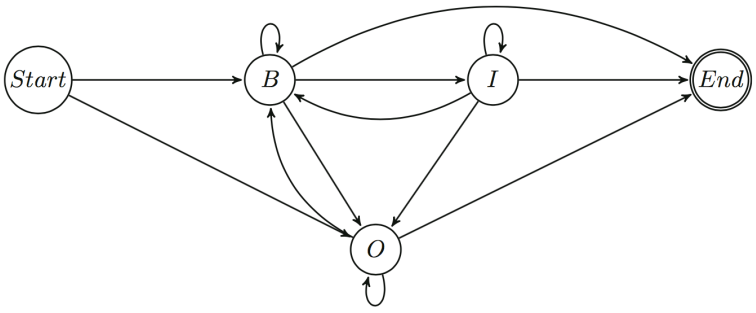

Figura 17. Representación gráfica de HMM (Mohit, 2014)

Siguiendo la representación, el proceso generativo es el siguiente:

1) La secuencia empieza en el nodo inicial, $Start$.

2) Para cada token, palabra, en la secuencia hay una probabilidad de transición de estado donde se decide la clase del token.
3) Tras cada transición, el estado destino genera una palabra.
4) La secuencia termina en el nodo final, End.

Para este proceso se deben definir dos conjuntos de parámetros para entrenar HMM:

- $P(y_i|y_{-i})$: La probabilidad de transición de estado es la probabilidad condicional de la etiqueta del token actual dada la etiqueta del token previo.
- $P(t_i|y_i)$: LA probabilidad de generar un token dada su etiqueta.

En el entrenamiento HMM aprende estos parámetros calculando la probabilidad de transiciones y generación de palabras en los datos de entrenamiento. Una vez finalizado el entrenamiento, se elige la etiqueta que maximiza el producto de los parámetros, este proceso se denomina decodificación y normalmente emplea el algoritmo de Viterbi.

Otros modelos de aprendizaje supervisado son: Máxima Entropía o Conditional Random Fields (CRF). Estos métodos no asumen la independencia de las palabras y sus etiquetas, permitiendo a los modelos beneficiarse de diversas características superpuestas.

2.13 Elección de modelo

En este proyecto se ha optado por el método hLDA debido tanto al conjunto de datos que se empleará en la validación, en el que la información de autores o fecha no son relevantes, como por la mayor complejidad del modelo que hace que un documento no tenga únicamente un tema, sino que se obtiene una estructura jerárquica de topics, permitiendo establecer relaciones semánticas más complejas entre dichos documentos.

3 ANTECEDENTES

El conocimiento empieza en el asombro

- Sócrates -

En capítulos anteriores se han introducido diversos métodos para resolver el problema de topic modeling. Todos ellos se engloban en los algoritmos de clustering de ML no paramétricos, en los que no se conoce previamente el número de parámetros y se emplean modelos de aproximación para escogerlos. En este capítulo se tratarán las técnicas empleadas en el algoritmo elegido, hLDA, desde la distribución que siguen los topics hasta el proceso de asignación de topics a las palabras.

3.1 Distribución de Dirichlet

La distribución Dirichlet se define como una distribución de probabilidad de números reales no negativos cuya suma es uno (Blei, Griffiths, & Jordan, 2010).

Sea U un vector aleatorio distribuido según una variable aleatoria Dirichlet donde α_i son los parámetros reales no negativos:

$$U \sim Dir(\alpha_1, \alpha_2, \dots, \alpha_K)$$

donde se cumple que:

$$\sum_{\Omega} P(U = u) = 1$$

$$P(U = u) \geq 0 \; \forall u$$

Estas condiciones hacen que la distribución de Dirichlet pueda verse como aquella distribución cuyos valores son funciones de masa de probabilidad.

Definición: Sea $V = [V_1, V_2, \dots, V_k]$ una variable aleatoria, V.A., donde $V_i > 0$ y $\sum_{k=1}^{K} V_k = 1$. Sea además $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_K]$ con $\alpha_i > 0$ y $\alpha_0 = \sum_{i=1}^{k} \alpha_i$. Entonces, se dice que V sigue una distribución de Dirichlet de parámetro α cuya función densidad de probabilidad viene dada por la expresión:

$$f(V, \alpha) = \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \prod_{i=1}^{K} V_i^{\alpha_i - 1}$$

La media de V_i se calcula empleando la siguiente expresión:

$$E[V_i] = \frac{\alpha_i}{\sum_{k=1}^{K} \alpha_k}$$

donde los valores de α_i determinan la concentración de V alrededor de la media. Se pueden distinguir tres casos:

- $\alpha_1 = \cdots = \alpha_K = 1$, la distribución es uniforme.
- $\alpha_i > 1$, se obtiene una distribución unimodal cuyo valor máximo se encuentra alrededor de la media
- $\alpha_i < 1$, da lugar a una distribución cuyo valor máximo se encuentra en los extremos.

En la siguiente figura se representan distintos resultados dependiendo de los valores de α. En el primer resultado nos encontramos en el tercer caso y puede apreciarse cómo los valores máximos se encuentran en los extremos. Los resultados intermedios corresponden al caso en el que $\alpha_i > 1$ y se observa cómo el valor máximo se encuentra en el valor medio. Por último, se representa el caso de la distribución uniforme.

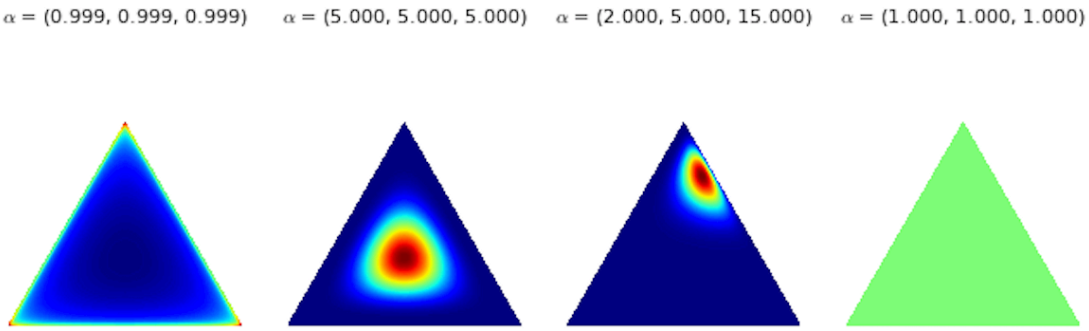

Figura 18. Distribución de Dirichlet para distintos valores de α.

Una generalización de la distribución de Dirichlet a infinitos conjuntos de eventos es el proceso de Dirichlet (DP) que es un proceso estocástico cuyos valores son distribuciones sobre un espacio muestra arbitrario. Este proceso se define por dos componentes: un parámetro de concentración, α, que especifica el grado de discretización del proceso, donde $\alpha \to 0$ significa que se concentra en un único valor y $\alpha \to \infty$, corresponde a una distribución continua; y la distribución base, H, que define los valores esperados del proceso, es decir, son los valores de la variable que sigue una determinada distribución sobre los que el proceso de Dirichlet define sus distribuciones en función de la probabilidad α. Mientras que H suele tomar valores continuos, El resultado del DP es discreto y su grado de discretización viene dado por α.

$$\big(G(A_1), \dots, G(A_k)\big) \sim Dir\big(\alpha H(A_1), \dots, \alpha H(A_k)\big) \to G \sim DP(\alpha, H)$$

donde $G(\cdot)$ es el proceso de Dirichlet.

Un ejemplo de proceso de Dirichlet es aquel en el que tenemos una urna donde se van insertando bolas de colores, donde el número de colores diferentes es infinito. Se empieza con la urna vacía y aleatoriamente se elige una bola de un color y se inserta en la urna. Posteriormente, hay dos opciones: coger aleatoriamente otra bola e insertarla en la urna, o extraer una bola de la urna e introducirla de nuevo en la urna junto a otra del mismo color. A medida que crece el número de bolas en la urna, la probabilidad de coger un nuevo color decrece. La proporción de bolas en la urna se define con el proceso de Dirichlet.

Para diferenciar entre una distribución y un proceso de Dirichlet se puede considerar el caso en el que se para a una persona en la calle y se le pregunta por su color favorito. Si limitamos sus opciones a seis colores posibles (negro, rosa, azul, verde, naranja y blanco), cada persona elegirá un color dependiendo de su estado de ánimo y esa elección puede modelarse como una función masa de probabilidad. Esto es una distribución de Dirichlet de parámetros los seis colores entre los que se puede elegir. Sin embargo, si no limitamos el conjunto de colores a escoger, para modelar la elección es necesario tener una distribución de distribuciones de espacio muestra infinito. Esto es el proceso de Dirichlet (Frigyik, Kapila, & Gupta, 2010).

Para emplear el proceso de Dirichlet en problemas en los que no conocemos el valor de los parámetros de entrada, o son infinitos, es necesario aplicar técnicas de muestreo como son los algoritmos Stick-Breaking Construction y Chinese Restaurant Process que se detallan en los siguientes apartados.

3.2 Stick-Breaking Construction

La distribución de Dirichlet usa un vector de K dimensiones cuyos componentes suman uno. El problema radica en que se debe conocer el valor de K. Sin embargo, en ocasiones, se quiere que este valor no esté

acotado. Por ejemplo, considerando el caso de la distribución de Dirichlet, donde se quiere obtener la distribución de una secuencia de números que puede verse a su vez como una distribución, nos encontramos en la situación de modelar una distribución de distribuciones. Para ello, se emplea el algoritmo conocido por *Stick-Breaking Construction* (SBC) (Blei, Griffiths, & Jordan, 2010).

Este algoritmo supone el intervalo (0,1) como un palo (stick) unitario y define $V_1 \sim Beta(\alpha_1, \alpha_2)$ como la primera fracción tras romper el palo. Sea $\theta_1 = V_1$ el primer fragmento del palo y $1 - \theta_1$, el resto de él, continuando el procedimiento de rotura del palo, se obtiene:

$$\theta_i = V_i \prod_{j=1}^{i-1} (1 - V_j)$$

donde $\{V_i\}$ es una secuencia infinita de variables independientes que siguen la distribución $Beta(\alpha_1, \alpha_2)$. En el caso $\alpha_1 = 1$, se obtiene la distribución GEM (Griffiths-Engen-McCloskey) que es un proceso estocástico de un parámetro, α_2. A mayor valor de α_2, el número de particiones es mayor. Cuanto menor sea α_2, menor será el palo para los valores posteriores (en promedio), lo que dará lugar a distribuciones más concentradas.

La siguiente figura muestra el histograma resultante de la ejecución del algoritmo SBC para un conjunto de 1000 muestras con función base $H = N(0,1)$. Se consideran dos casos: uno con valor de $\alpha_2 = 1$ y otro con $\alpha_2 = 50$. En el primer caso se observa cómo las particiones obtenidas se concentran en ciertos valores mientras que, en el segundo, se aprecia una mayor dispersión.

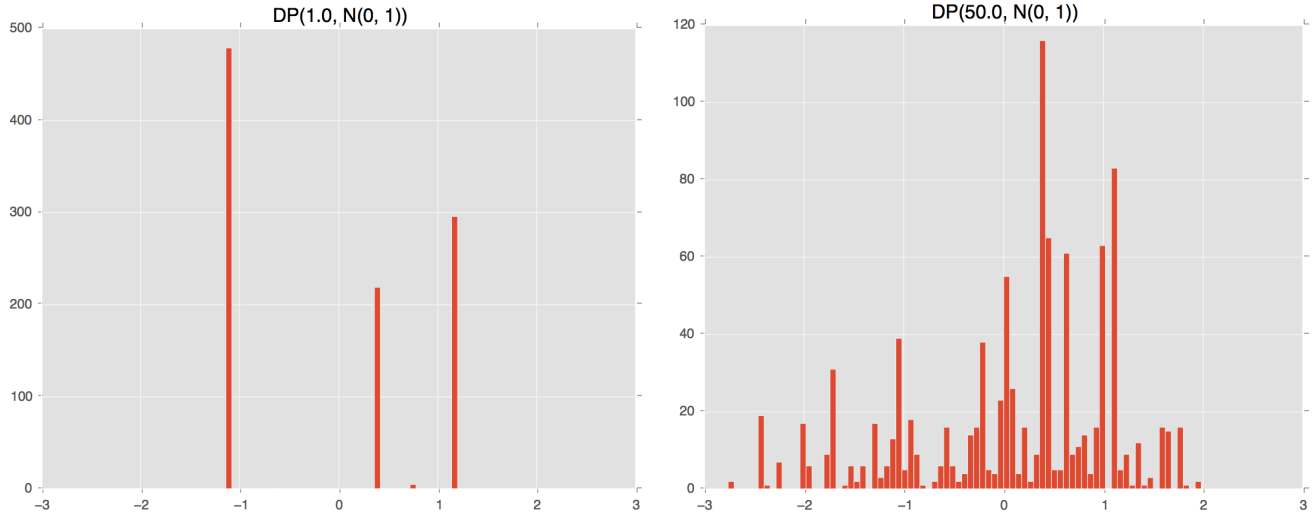

Figura 19. Histograma resultado de aplicación del algoritmo SBC. A la izquierda, con $\alpha_2 = 1$. A la derecha con $\alpha_2 = 50$. (Blei Lab, s.f.)

3.3 Chinese Restaurant Process

Es un algoritmo que permite trabajar con los datos sin conocer previamente el número de clusters, es decir, al igual que el método anterior, CRP busca extraer el valor de K a partir de los datos. *Trasladándolo al problema de topic modeling, CRP permitiría asignar documentos a topics sin conocer previamente el número de temas, sino aprendiéndolos de los datos*. Existen un número infinito de topics pero sólo un conjunto finito de éstos estará presente en los documentos (Gershman & Blei, 2011).

Se basa en el proceso estocástico de restaurante chino definido por Pitman y Dubins en 1983 (Aldous, 1985). Este proceso imagina la situación en la que N clientes quieren sentarse en un restaurante chino con infinitas mesas. El primer cliente se sienta en la primera mesa. El segundo cliente entra y se sienta en la primera mesa con probabilidad $\frac{1}{1+\gamma}$ y en la siguiente mesa libre con probabilidad $\frac{\gamma}{1+\gamma}$, donde γ, denominado parámetro de concentración es un número real y positivo que controla la frecuencia con la que un cliente escoge sentarse en una nueva mesa frente a una ya ocupada. Cuando llega el n-ésimo cliente al restaurante, se puede sentar en una

mesa ocupada con probabilidad proporcional al número de clientes anteriores o en una mesa desocupada con probabilidad proporcional a γ:

$$\begin{cases} p(mesa\ ocupada\ i\ |clientes\ anteriores) = \dfrac{n_i}{\gamma + n - 1} \\ p(siguiente\ mesa\ desocupada|clientes\ anteriores) = \dfrac{\gamma}{\gamma + n - 1} \end{cases}$$

donde n_i es el número de clientes que actualmente están sentados en la mesa i y n, el número de clientes que hay actualmente en el restaurante.

Se puede observar cómo a mayor valor de γ, mayor número de mesas ocupadas, es decir, menos clientes por mesa. *En el caso de topic modeling se traduciría en que habría un mayor número de topics y menos documentos en cada uno*. Por tanto, se concluye que los temas serían más específicos y, por tanto, habría menos documentos en cada uno.

La siguiente figura muestra un ejemplo de asignación de mesas con el algoritmo CRP.

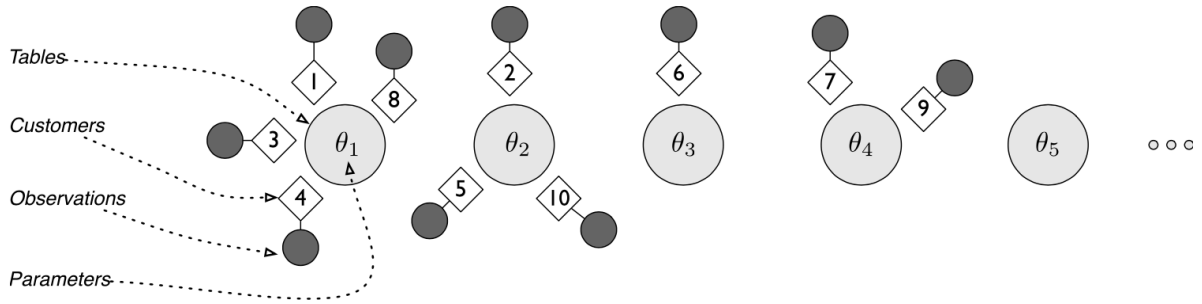

Figura 20. Representación de algoritmo CRP (Gershman & Blei, 2011)

Tras sentarse N clientes, tenemos una partición de N elementos cuya estructura es similar a la obtenida con los procesos de Dirichlet.

Si se asume que cada mesa se caracteriza por el parámetro θ_i que sigue una distribución, cada cliente se asociará al parámetro de la mesa en la que se siente. El resultado es una secuencia de estos parámetros que puede verse como un clustering probabilístico. Con el modelo CRP es posible determinar el número de clusters (número de mesas ocupadas) a partir de los datos y, posteriormente, permite asignar a estos clusters nuevos datos.

Este método es muy útil para los algoritmos de topic modeling, por ejemplo, LDA, debido a que se pueden establecer relaciones uno a uno entra las mesas y los componentes del algoritmo. Sin embargo, en modelos de jerarquía, como HLDA, se emplea una extensión de CRP, denominada *Nested Chinese Restaurant Process* (nCRP) debido a que cada dato se puede asociar a múltiples componentes de la jerarquía.

Una característica del modelo CRP es su intercambiabilidad de su distribución. Esto quiere decir que dicha distribución no depende del orden de llegada de los clientes. Esta propiedad es muy importante para el cálculo del posterior ya que permite el uso de técnicas de Monte Carlo para hacerlo más manejable.

3.4 Nested Chinese Restaurant Process

Como se ha mencionada anteriormente, es una extensión del algoritmo CRP que se emplea en el modelado de topics jerárquico.

nCRP se puede definir suponiendo el escenario donde hay un número infinito de restaurantes chinos de infinitas mesas en una ciudad. Uno de estos restaurantes será el nodo raíz y en cada una de sus infinitas mesas habrá una tarjeta con el nombre de otro restaurante. En todos los siguientes restaurantes sucederá lo mismo, tendrán en cada una de sus mesas una tarjeta de otro restaurante. Se supondrá que cada restaurante se refiere una única vez. Por tanto, los restaurantes de la ciudad se organizan en un árbol de infinitas ramas, donde cada uno de ellos se asocia a un nivel en el árbol. Cada vez que un cliente entra en un restaurante elige una mesa

siguiendo el proceso de CRP.

Un ejemplo es un turista que llega a una ciudad y en su primera noche entra en el restaurante raíz donde elige sentarse en una mesa determinada por CRP. La segunda noche se dirige al restaurante recomendado por la tarjeta de la mesa en la que se sentó y allí elige otra mesa. Esto se repite durante L días cuando el cliente tiene un conjunto de L restaurantes visitados que constituyen un trayecto desde el nodo raíz hasta el L-ésimo restaurante. Si este proceso lo repiten M turistas, el conjunto de trayectos forman un árbol de L niveles.

Esta distribución se emplea como prior en el modelado de topics jerárquico en el que cada documento describe uno de los trayectos del árbol.

La siguiente figura muestra la estructura en árbol del modelo nCRP donde cada nodo representa uno de los restaurantes. Puede apreciarse cómo cinco clientes llegan al primer restaurante y se sientan en determinadas mesas que los llevan a otro restaurante.

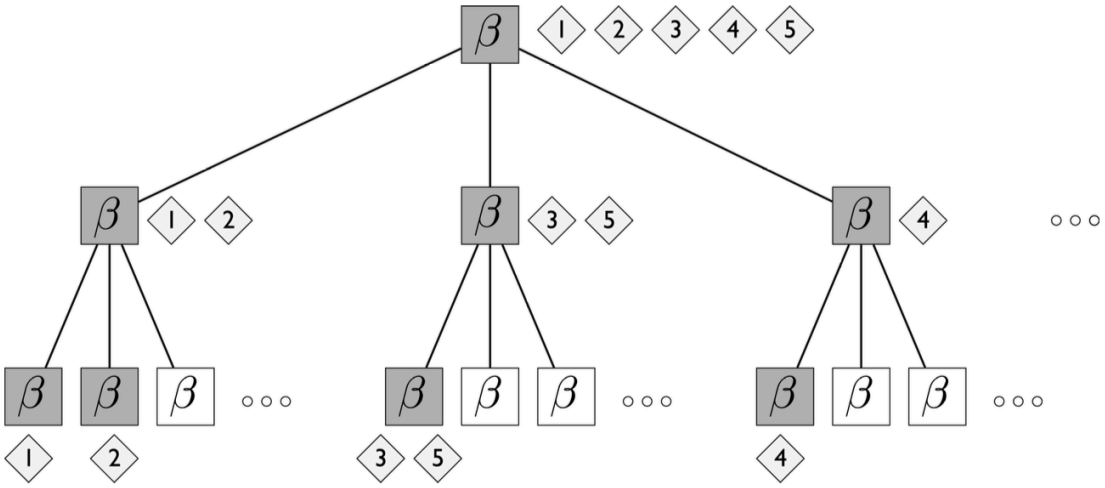

Figura 21. nCRP de tres niveles (Blei, Griffiths, & Jordan, 2010)

4 ALGORITMO IMPLEMENTADO. HLDA

Those who can imagine anything, can create the impossible

- Alan Turing -

En este capítulo se desarrollará el algoritmo implementado para resolver el problema de topic modeling. Ese modelo se denomina Hierarchical Latent Dirichlet Allocation, hLDA, y emplea los métodos descritos en el capítulo anterior para elaborar una estructura jerárquica en árbol con los topics de los documentos empleados como datos de entrada. Cada nodo se corresponde a un topic y cada documento poseerá tantos topics como nodos tenga su trayecto.

El algoritmo LDA definido en el Capítulo 2 define los topics como distribuciones de probabilidad sin relación entre ellos. Esto presenta el inconveniente de que se desconoce el nivel de abstracción de los topics así como la posible relación entre ellos.

Estos problemas se solucionan con el uso de nCRP, que sirve como base para el modelo hLDA. El algoritmo nCRP permite definir el prior en una topología en árbol sin limitar la ramificación o profundidad del árbol, de forma que los topics más genéricos se encuentran situados en los nodos más cercanos al raíz y los más específicos, alrededor de las hojas del árbol. Una vez definido el modelo se empleará una inferencia probabilística para identificar tanto los topics como las relaciones entre éstos.

Se considerará un conjunto de datos formado por un corpus de documentos. Cada documento es una colección de palabras y cada palabra forma parte de un vocabulario. hLDA asume que las palabras de un documento se han generado conforme a un modelo probabilístico.

Considerando la variable multinomial, z, y el conjunto de distribuciones asociadas a cada palabra, $p(w|z,\beta)$ donde β es un hiperparámetro. Este conjunto de distribuciones conformará los topics del documento. Cada documento pertenecerá a un topic con cierta probabilidad, θ. Temporalmente se asume un conjunto de topics finito, K. Por tanto, la variable z tomará valores entre los K posibles topics y θ será un vector de K dimensiones que define la probabilidad de pertenencia a cada topic.

Definidos estos parámetros, el algoritmo LDA se define como un proceso generativo de dos niveles en los que cada documento se asocia a cada topic con una determinada proporción y el corpus se modela como una distribución de Dirichlet de estas proporciones. Los dos niveles son:

1) Elegir un vector θ.
2) Para cada documento, muestrear palabras siguiendo la distribución $p(w|\theta)$ para el valor escogido de θ.

Puede apreciarse la independencia existente entre topics. Como alternativa, se propone el modelo hLDA donde se supone que se parte de un árbol de L niveles en el que cada nodo es un topic. Un documento se genera siguiendo este proceso:

1) Elegir un camino desde el nodo raíz al nodo hoja.
2) Estimar el vector de proporciones de topics θ a partir de la distribución de Dirichlet de L dimensiones.
3) Generar las palabras del documento a partir de los topics presentes en el camino desde el nodo raíz a la hoja.

Finalmente, se empleará el algoritmo nCRP para relajar la condición de estructura fija del árbol, calculando el prior. Un documento se genera primero eligiendo un camino del árbol de L niveles y se escogen las palabras atendiendo a los L topics asociados a dicho camino.

Formalmente, considerando el árbol infinito construido por nCRP y definiendo c_d como el camino trazado por el d-ésimo documento, el modelo hLDA tiene el siguiente proceso generativo:

Algoritmo hLDA

1) Para cada mesa del árbol infinito $k \in \mathcal{T}$:
   a) Estimar su topic asociado a partir de la distribución de Dirichlet: $\beta_k \sim Dir(\eta)$
2) Para cada documento del corpus $d \in \{1,2,\dots,D\}$:
   a) Elegir un camino del árbol siguiendo el algoritmo nCRP: $c_d \sim nCRP(\gamma)$
   b) Estimar la proporción de cada topic de su rama del árbol: $\theta_d|[m,\pi] \sim GEM(m,\pi)$
   c) Para cada palabra:
      i. Elegir un nivel/topic: $Z_{d,n} \mid \theta_d \sim Discrete(\theta_d)$
      ii. Elegir una palabra según el nivel/topic en el que se encuentre: $W_{d,n} \mid \{Z_{d,n}, c_d, \beta\} \sim Discrete(\beta_{c_d}[Z_{d,n}])$ que se encuentra parametrizado por el topic $Z_{d,n}$ del camino c_d

Cabe mencionar que los componentes probabilísticos son variables latentes, por ejemplo, los topics. Esto quiere decir que no se asumen unos topics predefinidos sino que se infieren estos valores a partir del posterior, condicionado al corpus de documentos y a las probabilidades de las variables latentes. Por ejemplo, suponiendo que d documentos han sido previamente generados, el documento $d+1$ puede seguir uno de los trayectos del árbol previamente definido o crear uno nuevo. Por tanto, el número de parámetros del modelo propuesto puede crecer, según el corpus. Esta flexibilidad permite optimizar el posterior para finalmente obtener el árbol que mejor se ajuste al conjunto de datos de entrada.

Sin embargo, existen otros parámetros que deben ser previamente definidos: el número de niveles del árbol o la forma son algunos ejemplos. Éstos se conocen como hiperparámetros. Por ejemplo, el parámetro de la distribución de Dirichlet, η, define la dispersión de los topics, es decir, a menor valor, los topics estarán formados por menos palabras lo que llevará a soluciones donde los árboles son más grandes.

Se debe considerar que el valor de estos parámetros depende en gran medida del corpus de documentos. Por tanto, es muy importante hacer un análisis con distintos valores hasta obtener aquellos que mejor se ajusten a los datos. En el siguiente capítulo se mostrarán los distintos resultados obtenidos para diversos valores de los hiperparámetros.

4.1 Inferencia probabilística

Tal y como se ha mencionado anteriormente, una vez seguido el proceso generativo de documentos, se deberá realizar la optimización del modelo hasta conseguir el árbol que mejor se ajuste a los datos, es decir, obtener el árbol que mejor estime la estructura de topics oculta en los documentos. Esta optimización se conoce como inferencia probabilística y empleará el posterior. Sin embargo, la expresión del posterior no resulta tratable y deben emplearse técnicas de aproximación. En el caso propuesto, se hará uso del *muestreo de Gibbs colapsado*, un algoritmo MCMC (Markov Chain Monte Carlo) en el que las variables latentes se muestrean iterativamente condicionadas a las observaciones y que marginaliza algunas de ellas para acelerar la convergencia. Se muestrearán los trayectos por documentos y las asignaciones de topics por nivel y se marginalizarán los parámetros de Dirichlet, β_k, y de proporciones de topics, θ_d. Con esto se aproximará el posterior $p(c_{1:D}, z_{1:D}|\gamma, \eta, m, \pi, w_{1:D})$ donde γ define la tendencia de los clientes de compartir mesa en cada restaurante, η refleja la varianza de topics y m y π, la asignación de palabras en los distintos niveles.

El algoritmo se divide en dos partes principales: el muestreo de asignación de niveles y el muestreo de asignación de trayectos.

4.1.1 Muestreo de asignación de niveles

Dada una elección de trayecto, se deberá muestrear la asignación de la variable de asignación de topics por niveles, $z_{d,n}$, que para la palabra n del documento d sigue la expresión:

$$p(z_{d,n}|z_{-(d,n)}, c, w, m, \pi, \eta) \propto p(z_{d,n}|z_{d,-n}, m, \pi)p(w_{d,n}|z, c, w_{-(d,n)}, \eta)$$

donde $z_{-(d,n)}$ y $w_{-(d,n)}$ son los vectores de asignación de niveles y de palabras observadas, extrayendo $z_{d,n}$ y $w_{d,n}$, respectivamente.

La primera parte de la expresión es una distribución de niveles que, al tener un número infinito de componentes, se muestrea por etapas. Primero se muestrea de una distribución de todo el espacio de niveles presente en el documento y un nivel más profundo al actual.

$$\begin{aligned} p(z_{d,n} = k|z_{d,-n}, m, \pi) &= E\left[V_k \prod_{j=1}^{k-1}(1 - V_j)|z_{d,-n}, m, \pi\right] \\ &= E[V_k|z_{d,-n}, m, \pi]\prod_{j=1}^{k-1} E[(1 - V_j)|z_{d,-n}, m, \pi \\ &= \frac{m\pi + \#[z_{d,-n} = k]}{\pi + \#[z_{d,-n} \geq k]}\prod_{j=1}^{k-1}\frac{(1-m)\pi + \#[z_{d,-n} > j]}{\pi + \#[z_{d,-n} \geq j]} \end{aligned}$$

donde $\#[\cdot]$ cuenta los elementos que cumplen determinada condición.

La segunda parte de la expresión es la probabilidad de una palabra basado en su asignación.

$$p(w_{d,n}|z, c, w_{-(d,n)}, \eta) \propto \#[z_{-(d,n)} = z_{d,n}, c_{z_{d,n}} = c_{d,z_{d,n}}, w_{-(d,n)} = w_{d,n}] + \eta$$

4.1.2 Muestreo de trayecto

Dadas las variables de elección de nivel, se muestrea el trayecto asociado a cada documento condicionado a los otros trayectos y palabras observadas. Para ellos, la expresión que se sigue es:

$$p(\boldsymbol{c}_d|\boldsymbol{w}, \boldsymbol{c}_{-d}, \boldsymbol{z}, \eta, \gamma) \propto p(\boldsymbol{c}_d|\boldsymbol{c}_{-d}, \gamma)p(\boldsymbol{w}_d|\boldsymbol{c}, \boldsymbol{w}_{-d}, \boldsymbol{z}, \eta)$$

Esta expresión puede verse como una instancia del teorema de Bayes donde $p(\boldsymbol{w}_d|\boldsymbol{c}, \boldsymbol{w}_{-d}, \boldsymbol{z}, \eta)$ es la probabilidad de una palabra dada una asignación de trayecto y $p(\boldsymbol{c}_d|\boldsymbol{c}_{-d}, \gamma)$ es el prior de trayectos definido con nCRP. La probabilidad de los datos se obtiene integrando los parámetros multinomiales, derivando en la siguiente expresión:

$$\begin{aligned} &p(\boldsymbol{w}_d|\boldsymbol{c}, \boldsymbol{w}_{-d}, \boldsymbol{z}, \eta) \\ &= \prod_{l=1}^{\max(z_d)} \frac{\Gamma(\sum_w \#[\boldsymbol{z}_{-d} = l, \boldsymbol{c}_{-d,l} = c_{d,l}, \boldsymbol{w}_{-d} = w] + V\eta)}{\prod_w \Gamma(\#[\boldsymbol{z}_{-d} = l, \boldsymbol{c}_{-d,l} = c_{d,l}, \boldsymbol{w}_{-d} = w] + \eta)} \frac{\prod_w \Gamma(\#[\boldsymbol{z} = l, \boldsymbol{c}_l = c_{d,l}, \boldsymbol{w} = w] + \eta)}{\Gamma(\sum_w \#[\boldsymbol{z} = l, \boldsymbol{c}_l = c_{d,l}, \boldsymbol{w} = w] + V\eta)} \end{aligned}$$

Se debe tener en cuenta que cada trayecto de define en bloque debido a que su valor en cada nivel depende del anterior.

4.1.3 Algoritmo de muestreo de Gibbs

Una vez definidos los pasos a seguir, el algoritmo de muestreo de Gibbs se resume en: dado el estado actual, $\left\{c_{1:D}^{(t)}, z_{1:D}^{(t)}\right\}$, se muestrea iterativamente cada variable condicionada al resto:

Algoritmo de muestreo de Gibbs

1) Para cada documento del corpus $d \in \{1,2,\dots,D\}$:
   a) Estimar aleatoriamente $c_d^{(t+1)}$ empleando:
   $$p(\boldsymbol{c}_d|\boldsymbol{w},\boldsymbol{c}_{-d},\boldsymbol{z},\eta,\gamma) \propto p(\boldsymbol{c}_d|\boldsymbol{c}_{-d},\gamma)p(\boldsymbol{w}_d|\boldsymbol{c},\boldsymbol{w}_{-d},\boldsymbol{z},\eta$$
   b) Estimar aleatoriamente $z_{n,d}^{(t+1)}$ empleando:
   $p(z_{d,n}|z_{-(d,n)},c,w,m,\pi,\eta) \propto p(z_{d,n}|z_{d,-n},m,\pi)p(w_{d,n}|z,c,w_{-(d,n)},\eta)$ para cada palabra $n \in \{1,2,\dots,N_d\}$

La distribución estacionaria de la cadena Markov correspondiente es la distribución condicional de las variables latentes en el modelo hLDA dado el corpus. Tras varias iteraciones del algoritmo se alcanza una distribución estacionaria y se comienza a extraer muestras en determinados intervalos seleccionados para minimizar la autocorrelación y aproximar el verdadero posterior con la distribución empírica correspondiente.

4.2 Implementación del algoritmo

Una vez definido el modelo implementado, en este apartado se presenta la estructura del código. El lenguaje de programación elegido para la implementación ha sido Python debido a la multitud de librerías de las que dispone y al ser un lenguaje que ha ido cobrando gran importancia en los últimos años.

Las librerías que se han empleado en el código han sido:

1. `numpy`. Importa operaciones matemáticas.
2. `csv`. Se emplea para leer/escribir ficheros .csv.
3. `math`. Contiene funciones matemáticas. De aquí se importará la función `log`.
4. `sys`. Se empleará en la salida por línea de comandos de los resultados.
5. `re`. Se empleará para el uso de expresiones regulares. Para transformar el vocabulario de la base de datos en un formato adecuado para su posterior procesado.
6. `scipy`. Para importar además de las operaciones de `numpy`, otras estadísticas.

Este código se estructura en varias funciones presentadas en la siguiente tabla.

Interfaz	Descripción
`main()`	Función principal del programa. En ella se llaman al resto de funciones.
`convert_doc_to_list(doc_index, doc_words)`	Esta función se emplea para construir el corpus de documentos con el formato adecuado.
`load_vocab(vocab_file)`	Esta función carga el vocabulario empleado en el corpus de documentos.
`load_corpus(corpus_file)`	Esta función carga el corpus de documentos y lo transforma hasta conseguir el formato adecuado para el algoritmo.
`HierarchicalLDA(corpus, vocab, alpha, gamma, eta, num_levels)`	Clase donde se definen las distintas funciones que se emplearán en el algoritmo hLDA.

`nCRP_node(num_levels, vocab, parent, level)`	Clase que define a un nodo del árbol resultante del algoritmo nCRP.

Todas estas funciones y clases han sido desarrolladas en este proyecto, excepto `convert_doc_to_list(doc_index, doc_words)` que se ha reutilizado de un código de libre acceso el cuál hace uso de una de las bases de datos empleadas en este proyecto (GitHub, s.f.).

La siguiente figura muestra el diagrama de flujo de la función principal main().

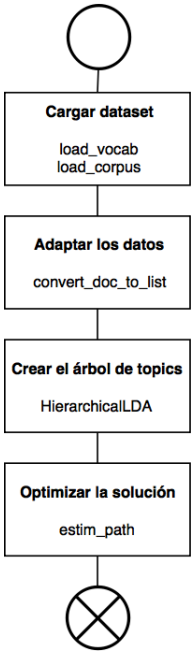

Figura 22. Diagrama de flujo de main()

4.2.1 nCRP_node

Es una clase donde se definen las distintas funciones que se emplean en la creación del árbol de topics. Estas funciones son:

1. `add_child`. Añade un nuevo nodo al árbol.
2. `check_leaf`. Comprueba si un nodo es una hoja o no, es decir, si pertenece o no al último nivel.
3. `add_new_path`. Añade nodos hasta llegar a la hoja (último nivel).
4. `remove_path`. Elimina un documento de un camino.
5. `remove_child`. Elimina un nodo hijo.
6. `add_to_path`. Añade un documento el camino.
7. `select_node`. Elegir un nodo o crear uno nuevo cuando llega un nuevo documento. Base del algoritmo nCRP.
8. `get_top_words`. Obtiene las palabras más usadas en un topic.

4.2.2 HierarchicalLDA

Es una clase donde se definen las distintas funciones que se emplean en el algoritmo hLDA. Este algoritmo se estructura en diversas partes:

1. Inicializar un camino, fijando un nodo raíz.
2. Para cada documento del corpus, recorrer cada nivel e ir fijando su camino en el árbol de topics. Para ello se llamará a la función `select_node` definida en el apartado anterior. Finalmente, se recorre cada palabra del documento y se asigna a un nodo de su camino.
3. Estimación y optimización de la solución aplicando el muestreo de Gibbs.

La Figura 23 muestra el diagrama de flujo de la clase HierarchicalLDA, donde se inicializan variables y se forma el árbol de topics.

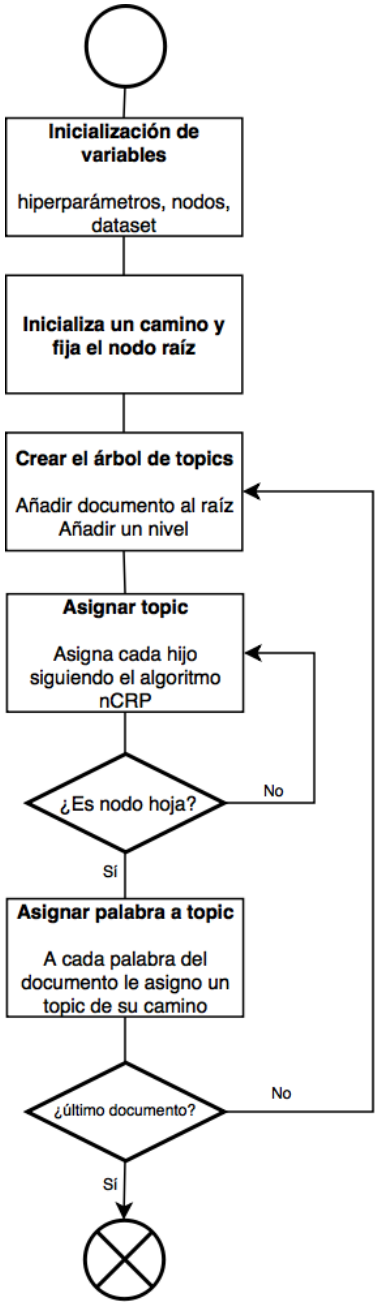

Figura 23. Diagrama de flujo de la creación del árbol de topics

Dentro de esta clase se definen a su vez otras funciones que se emplean en la parte de estimación y optimización del algoritmo. Estas funciones son:

1. `estim_path`. En esta función se implementa el algoritmo de muestreo de Gibbs, donde se recorre cada documento y se muestrea el camino, llamando a la función `sample_path`. Finalmente, se muestrea el topic asociado a cada palabra del documento haciendo uso de la función

`sample_topics`.

2. `sample_path`. Define el camino que sigue un documento desde el nodo raíz hasta la hoja. Calcula la aproximación de la verosimilitud para esa solución.
3. `calculate_prior`. Calcula el prior del camino siguiendo el algoritmo nCRP.
4. `calculate_doc_likelihood`. Calcula la verosimilitud de un documento.
5. `calculate_word_likelihood`. Calcula la verosimilitud de una palabra, siguiendo la ecuación definida en apartados anteriores.
6. `sample_topics`. Muestrea el topic al que pertenece cada palabra del documento.

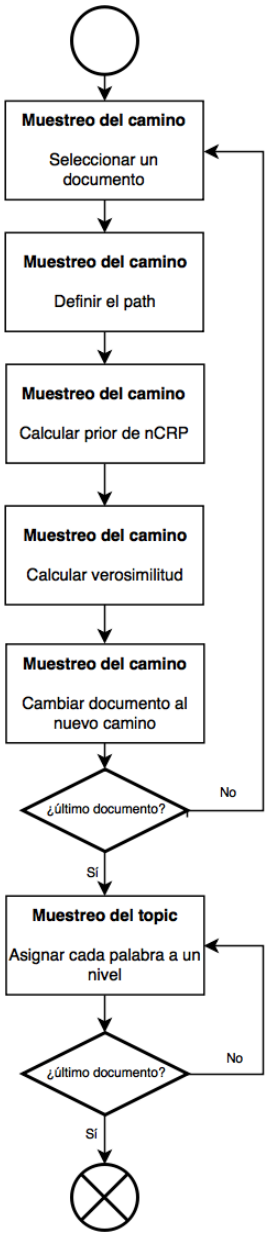

Figura 24. Diagrama de flujo de la estimación del camino

Por último, se definen dos funciones que se emplean en la representación de los resultados:

1. `print_node`. Imprime la información de cada nivel: topic, número de documentos en el topic, palabras más usadas en el topic y el número de repeticiones de dichas palabras.
2. `print_nodes`. Se emplea en la función `estim_path` y llama a su vez `la print_node` para mostrar por pantalla los resultados del algoritmo.

5 RESULTADOS. ANÁLISIS DE PARÁMETROS

Intelligence is the ability to adapt to change

- Stephen Hawking -

En este capítulo se mostrarán los resultados obtenidos con el algoritmo implementado así como la realización de un análisis de los hiperparámetros de éste.

Para validar el modelo propuesto se han empleado dos bases de datos diferentes: la primera de ellas se corresponde con un conjunto de 1500 papers de *Sistemas de procesado de información neuronal* (Neural Information Processing System – NIPS) (Lichman, 2013) y la segunda de ellas, CORA (Andrew McCallum, s.f.), se compone de un conjunto de más de 11000 papers de 80 topics diferentes. *Es muy importante tener en cuenta que, debido a la diferencia de documentos en ambos corpus, los parámetros óptimos serán diferentes para cada uno*. Además, dependerá en gran medida del tipo de análisis que se busque (una clasificación más general o una más restrictiva).

Todos estos aspectos deberán ser considerados por el usuario a la hora de ejecutar el algoritmo.

Los hiperparámetros que se analizarán son: α, γ y η. Además, se deberá elegir el número de niveles que se desea emplear en el árbol.

En la siguiente tabla se resume el conjunto de experimentos que se han llevado a cabo para la validación del algoritmo.

Base de datos	Niveles	α	γ	η	Caso de estudio
NIPS	50	2	1	2	1
NIPS	50	1	1	2	2
NIPS	50	0.5	1	2	3
NIPS	50	0.2	1	2	4
NIPS	50	1	2	2	5
NIPS	50	1	0.5	2	6
NIPS	50	1	1	1	7
NIPS	50	1	1	0.5	8
CORA	10/20/30/50	-	-	-	1 - 8

Tabla 1. Conjunto de experimentos para la validación del algoritmo

Para la base de datos CORA, se emplearán un subconjunto de 175 documentos y se repetirán los 8 casos de estudio definidos para distintos valores de niveles.

5.1 Base de datos NIPS

En este apartado se analizará la primera de las bases de datos previamente citadas. Este corpus se compone de 1500 papers y el número de niveles del árbol de topics empleado para su estudio ha sido fijado en 50, por tanto, un documento tendrá, como máximo, 50 topics. A continuación, se mostrarán los resultados obtenidos al variar los hiperparámentros del algoritmo y se extraerán algunas conclusiones.

5.1.1 Análisis del hiperparámetro α

En los siguientes subapartados se mostrarán los resultados obtenidos para distintos valores del hiperparámetro α. Para ello se fijará el resto de variables. Debido a la gran extensión de los resultados, en este capítulo se muestran una parte de éstos de forma que se pueda realizar una comparativa.

5.1.1.1 Caso de estudio 1

En este primer caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
2	1	2

Tabla 2. Hiperparámetros del caso de estudio 1

Los resultados obtenidos en este caso se resumen en la siguiente tabla. Se ha de tener en cuenta que, debido a la gran extensión de los datos, se ha representado únicamente algunas ramas del árbol de topics devuelto por el algoritmo. En el Anexo se encuentra una gráfica que complementa a estos resultados.

Nivel	Rama	Topic	Número de documentos	Palabras más empleadas
0	0	0	1499	Recurrently, stringent, symbolic, sequencing, grammatical, finitely, sequencer, static, translate, languages
1 - 37	0	1 - 3032	1499	Circuitry, analogical, modeled, database, distributional, functional, methologies, classifies, learnt, algorithmic
38 - 44	1	3033 - 3039	1498	Speechreading, recognizable, systematic, speaking, visualization, imaginary, motivate, connectivity, representational, directional
45	1.1	3040	1245	Cost, accent, OSPF, ZFE, efficient, run, due, schemes
46	1.1.1	3041	408	Explanatory, promote, cyclic, inserted, sand, viewed, readily, encouraging
47	1.1.1.1	3041	29	Scaled, persist, bumptrees, altogether, static, guaranteed, vanishing
48	1.1.1.1.1	61545	20	Goodness, robertson, cooling, leonard, dash, gener, render
49	1.1.1.1.1.1	61546	12	Davies, michalski, johnson, richardson, petersen,

				indexed, autistic, thompson, erkki, williamson
49	1.1.1.1.1.2	198175	4	postdoctoral, subunit
49	1.1.1.1.1.3	198629	4	smoothing

Tabla 3. Resultados del caso de estudio 1

Finalmente, en la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 37	1
38 - 44	2 (uno de los cuales no tiene coincidencia de palabras, por tanto, llegará hasta el nivel 37)
45	4 (uno de los cuales no tiene coincidencia de palabras, por tanto, llegará hasta el nivel 44)
46	7 (uno de los cuales no tiene coincidencia de palabras, por tanto, llegará hasta el nivel 45)
47	33 (uno de los cuales no tiene coincidencia de palabras, por tanto, llegará hasta el nivel 46)
48	149 (diez de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 47)
49	517 (63 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 48)

Tabla 4. Resumen de los resultados del caso de estudio 1

La *Figura 25* representa algunas de las ramas del árbol de topics definido con estos parámetros. El código de colores de los niveles se resume en la siguiente tabla:

Color	Nivel
Celeste	0
Amarillo	1 - 37
Rojo	38 - 44
Verde	45
Rosa	46
Lila	47
Naranja	48
Añil	49

Tabla 5. Código de colores del árbol de topics

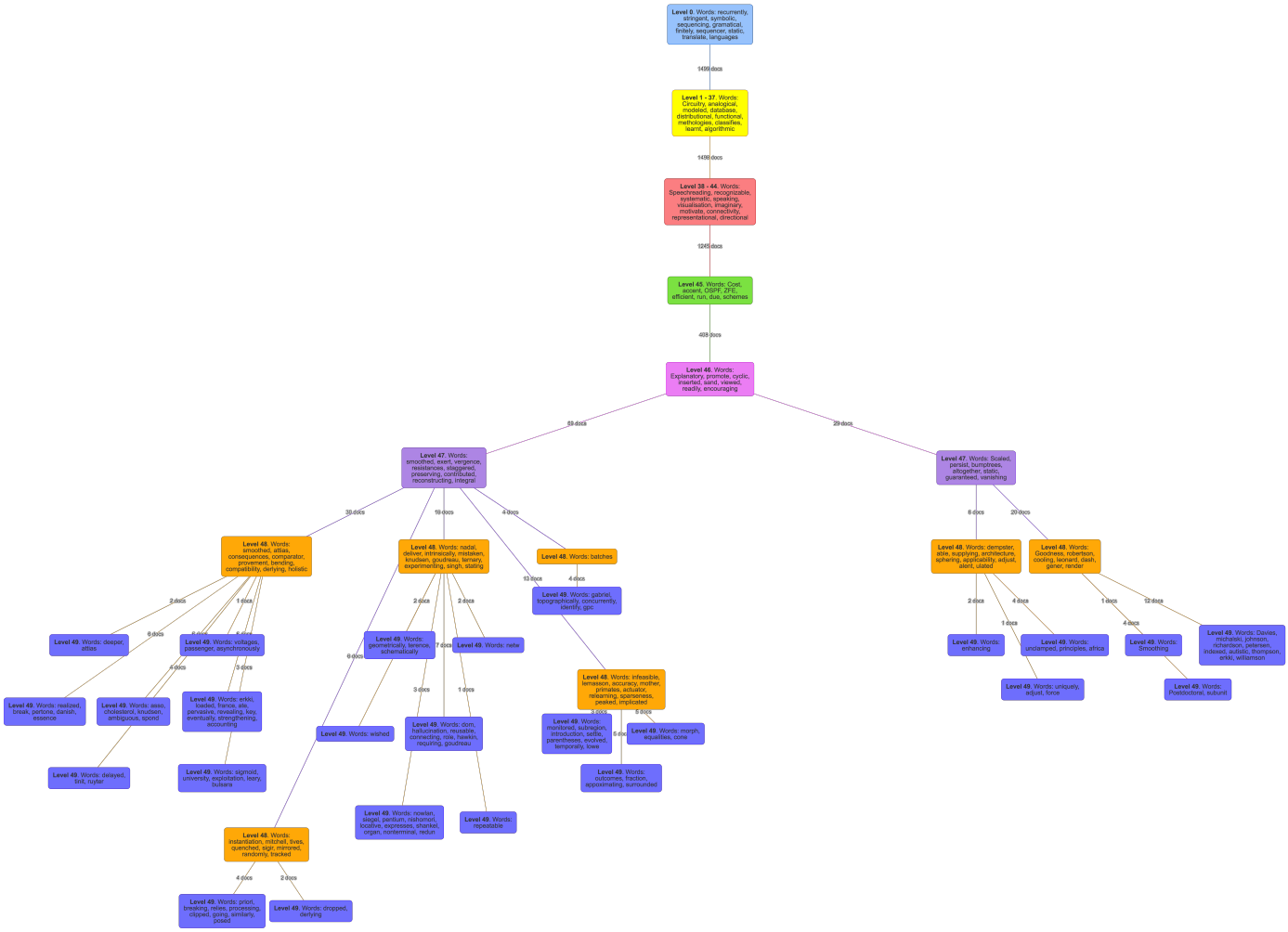

Figura 25. Representación parcial del árbol de topics resultante del caso de estudio 1

5.1.1.2 Caso de estudio 2

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
1	1	2

Tabla 6. Hiperparámetros del caso de estudio 2

Los resultados obtenidos en este caso se resumen en la siguiente tabla. Al igual que en el apartado anterior, se ha representado únicamente algunas ramas del árbol de topics devuelto por el algoritmo.

Nivel	Rama	Topic	Número de documentos	Palabras más empleadas
0	0	0	1499	psaltis, obermayer, brainard, hetero, pruning, hastad, retrieval, damaged, surgery, storm
1 - 44	0	98 - 53292	1499	Network, returned, codeword, genetically, learnt, problematic, systematic, analogical, objection, modeled,

				characterised, database, dynamical
45	1	53293	1498	functional, boundaries, theoretic, thresholded, prop, polynomially, resultant, dimensional, numbered, cases
46	1.1	53294	294	inferred, hoeffding, dure, mori, soon, masses, andersen, flying, expectancy, cycle
47	1.1.1	53295	39	unity, polar, selecting, column, importance, higgin, additively, ism, mathematically, victimized
48	1.1.1.1	53296	26	opposite, mismatches, leonard, linearize, feeding, award, potentiation, oshima, exploitation, graceful
49	1.1.1.1.1	62582	10	Davies, michalski, johnson, richardson, petersen, indexed, autistic, thompson, erkki, williamson
49	1.1.1.1.2	380756	8	Fop, form, fork, forgetting, forget, forgery, forgeries, forever, forest ** En este caso ninguna palabra coincide con el texto. Esto significa que esos ocho documentos tendrían 48 topics*
49	1.1.1.1.3	382178	2	Heiligenberg, movshon
49	1.1.1.1.4	382242	3	Cooperate
49	1.1.1.1.5	382251	3	Gradual, clin, variable
48	1.1.1.2	381991	3	Ease, watching, iir
49	1.1.1.2.1	382378	3	Liung, ply, cardiac
49	1.1.2.2.1	382095	2	Traiec, propagates
49	1.1.2.2.2	382677	2	gilloux

Tabla 7. Resultados del caso de estudio 2

Finalmente, en la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 44	1
45	2
46	7
47	55 (dos de los cuales no tienen coincidencia de palabras, por tanto, se quedan en el nivel 46)
48	241 (34 de los cuales no tienen coincidencia de palabras, por tanto, se quedan en el nivel 47)

49	696 (231 de los cuales no tienen coincidencia de palabras, por tanto, se quedan en el nivel 48)

Tabla 8. Resumen de los resultados del caso de estudio 2

5.1.1.3 Caso de estudio 3

En este tercer caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
0.5	1	2

Tabla 9. Hiperparámetros del caso de estudio 3

Los resultados correspondientes a algunas ramas del árbol obtenidos en este caso se resumen en la siguiente tabla.

Nivel	Rama	Topic	Número de documentos	Palabras más empleadas
0	0	0	1499	studentship, teaching, overlapped, committees, replicas, physic, ensembles, biasing, phased, sabes
1 - 44	0	379 - 561	1499	Bootstrapping, visualisation, modeled, cellular, place, directional, responses, eyes, boundaries, vectorial, problematic, resultant, functional, algorithmic, learnt
45	1	562	525	translates, parallelizable, visit, movie, learn, hornik, prototypes, analysis, puddle, france
46	1.1	563	278	davies, johnson, jan, thompson, geometric, alleviate, michalski, manipulated, rohwer, meador
47	1.1.1	564	196	learnt, traced, ilk, dates, semiparametric, continuing, laughton, tissues, regularizer, nature
48	1.1.1.1	565	189	asa, multimodal, obscure, parcor, ternational, flexibility, rounded, system, ibl, esti
49	1.1.1.1.1	566	169	circuitry, chir, analogical, curried, vlvp, outseg, voltages, neuronal, implemented, systematic
49	1.1.1.1.2	665086	7	** En este caso ninguna palabra coincide con el texto. Esto significa que esos siete documentos tendrían 48 topics*
49	1.1.1.1.3	666106	4	thomas, rub, tinmann
49	1.1.1.1.4	666832	2	quantization, competitor
49	1.1.1.1.5	666884	7	trivially, structuring, detectable, nonparametric

48	1.1.1.2	662090	4	multilayered, simplified, cup, sized, restore, memoryless, blind, reasoning, plausible
49	1.1.1.2.1	666482	1	** En este caso ninguna palabra coincide con el texto. Esto significa que ese documento tendría 48 topics*
49	1.1.1.2.2	666753	3	composed, going

Tabla 10. Resultados del caso de estudio 3

Finalmente, en la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 44	1
45	7 (1 de los cuales no tiene coincidencia de palabras. Por tanto, esos documentos se quedan en el nivel 44)
46	34 (2 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 45)
47	135 (45 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 46)
48	391(182 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 47)
49	804 (444 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 48)

Tabla 11. Resumen de los resultados del caso de estudio 3

5.1.1.4 Caso de estudio 4

Los valores de los hiperparámetros analizados en este caso son los siguientes:

α	γ	η
0.2	1	2

Tabla 12. Hiperparámetros del caso de estudio 4

Los resultados obtenidos en este caso se resumen en la siguiente tabla. Al igual que en el apartado anterior, se ha representado únicamente algunas ramas del árbol de topics devuelto por el algoritmo.

Nivel	Rama	Topic	Número de documentos	Palabras más empleadas
0	0	0	1499	circuitry, chir, analogical, curried, outseg, vlvp, voltages, neuronal, systematic, implemented
1 - 33	0	97 - 21008	1499	Dynamical, pointed, functional, algorithmic, learnt, controllable, problematic, optimally, systematic, methologies, parameterised, distributional, database, modeled, boundaries,

				objection, cellular
34 - 40	1	21010-21016	1498	Genetically, poral, played, fitted, classifies, neighborhood, decisive, gamma, gramatical, representational, symbolic, stringent, languages
41	1.1	21017	1497	entire, phased, autoassociation, variance, legal, burke, breast, complex, scale, sram
42	1.1.1	21018	1492	book, essential, systematic, jude, recorded, agglomerative, ima, cybern, similarities, helpful
43	1.1.1.1	21019	1045	arrangement, structuring, entries, recommendation, simard, commanded, carry, similarly, comfort, dealt
44	1.1.1.1.1	21020	239	improved, nonterminal, green, diffusive, postsubiculum, home, highlight, decides, crosscorrelation
45	1.1.1.1.1.1	21021	78	psychologically, trajec, compartment, uncorrelated, calif, harder, supported, competence, shaped, biological
46	1.1.1.1.1.1.1	21022	62	modules, exclusive, official, pendulum, york, lwpcr, explanatory, sirosh, som
47	1.1.1.1.1.1.1.1	21023	45	hikaridai, intelli, appealing, iteratively, generafive, invest, subsequence, settle, pursuing, redefine
48	1.1.1.1.1.1.1.1.1	21024	13	shapley, freq, nel, sardnet
49	1.1.1.1.1.1.1.1.1.1	62299	11	davies, michalski, johnson, richardson, petersen, autistic, indexed, thompson, erkki, williamson
49	1.1.1.1.1.1.1.1.1.2	1598221	2	** En este caso ninguna palabra coincide con el texto. Esto significa que esos dos documentos tendrían 48 topics*
48	1.1.1.1.1.1.1.1.2	1589571	31	** En este caso ninguna palabra coincide con el texto. Esto significa que esos 31 documentos tendrían 47 topics*
48	1.1.1.1.1.1.1.1.3	1596164	1	metrical, stably

Tabla 13. Resultados del caso de estudio 4

Finalmente, en la siguiente tabla se muestran el número de niveles obtenidos en este caso de estudio para realizar comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 33	1

34 - 40	2 (1 de los cuales no tiene coincidencia de palabras. Por tanto, ese documento se queda en el nivel 33)
41	3 (2 de los cuales no tiene coincidencia de palabras. Por tanto, esos documentos se quedan en el nivel 40)
42	4 (3 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 41)
43	8 (3 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 42)
44	17 (8 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 43)
45	59 (18 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 44)
46	173 (90 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 45)
47	410 (282 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 46)
48	746 (597 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 47)
49	1101 (903 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 48)

Tabla 14. Resumen de los resultados del caso de estudio 4

5.1.1.5 Conclusiones tras la variación del hiperparámetro α

Tras la realización de diversos experimentos, cambiando el valor del hiperparámetro α, se pueden llegar a las siguientes conclusiones:

1) A mayor valor de α, hay menor número de niveles más altos. Esto quiere decir que la clasificación es menos selectiva y se centra en extraer temas más genéricos o comunes a un mayor número de documentos.
2) En el caso de una clasificación más selectiva, es decir, menores valores de α, hay una mayor variedad de topics. Sin embargo, se ha de considerar que el número de documentos que se quedan en niveles más altos es mayor. Esto se puede traducir a que, debido al carácter específico de los temas, hay muy pocos documentos que cumplan esos criterios de pertenencia.

La siguiente figura muestra una comparativa de los casos de estudio del hiperparámetro α. Cada columna representa el número de topics diferentes existentes en cada nivel, cada topic puede tener uno o más documentos. Por ejemplo, el caso de estudio 4 posee 41 topics distintos en el nivel 45 del árbol.

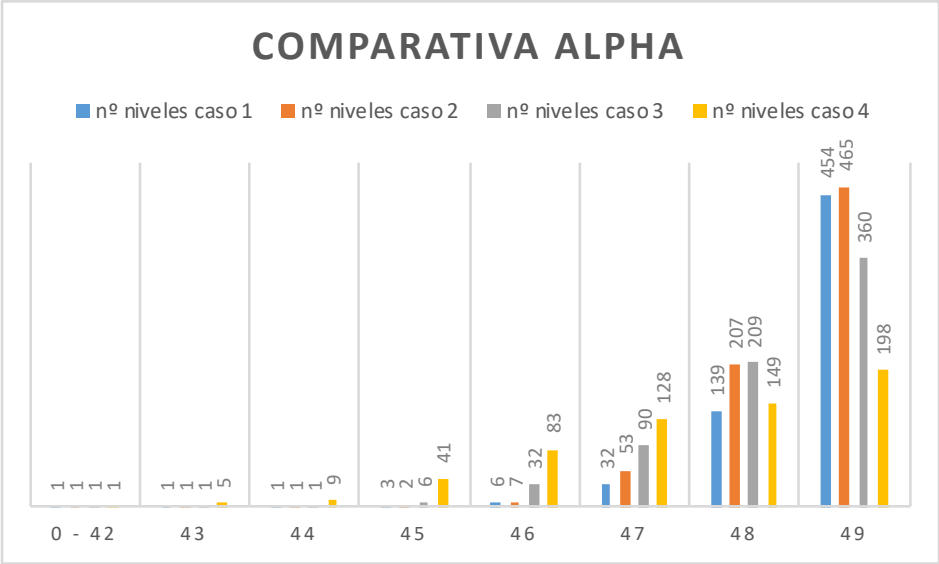

Figura 26. Gráfica comparativa de distintos valores del hiperparámetro α

5.1.2 Análisis del hiperparámetro γ

En este apartado se llevará a cabo un estudio del hiperparámetro γ presente en el algoritmo implementado. En este caso, esta variable afecta al método nCRP, encargado de la formación del árbol de topics.

En los siguientes sub-apartados se presentan los diversos casos de estudio.

5.1.2.1 Caso de estudio 5

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
1	2	2

Tabla 15. Hiperparámetros del caso de estudio 5

Los resultados obtenidos en este caso se resumen en la siguiente tabla. Al igual que en el apartado anterior, se ha representado únicamente algunas ramas del árbol de topics devuelto por el algoritmo.

Nivel	Rama	Topic	Número de documentos	Palabras más empleadas
0	0	0	1499	proctolin, blocked, instructive, parallelism, built, stark, simi, communities, operational, bob
1 - 31	0	50 - 53160	1499	Stringent, symbolic, gramatical, languages, trees, known, contextual, recognizable, modeled, inquiry, learnt, neurally, unitary, analogical, visualisation, objection, features, characterised
32 - 39	1	53161-53168	1498	Prototypical, fundamentally, database, algorithmic, functional, parameterised,

				methologies, boundaries, theoretic, resultant
40 - 45	1.1	53169-53174	1497	Labelling, stressed, syllables, templates, learnable, shared, marking, primate, perpendicular, classifies, decisive
46	1.1.1	53175	1062	styles, context, bilistic, stylistic, median, illegal, asymmetrical
47	1.1.1.1	53176	221	spaces, proving, angles, numerous, patrice, dependences, run, entirely, neurophysiol, brauer
48	1.1.1.1.1	53177	125	Anand, rotating, parametrization, regard, illustrates, ret, hrtf, simmon, actuator, restart
49	1.1.1.1.1.1	53178	107	controllable, modeled, motorola, moves, controlling, arma, inversely, tral, foster, dynamical
49	1.1.1.1.1.2	308497	4	believe, innate, testable, severity, eta
49	1.1.1.1.1.3	310200	4	prototypes, graz, maryland, frg, generically, string, templates, noiseless
49	1.1.1.1.1.4	312111	4	gang, addemup
49	1.1.1.1.1.5	312258	3	** En este caso ninguna palabra coincide con el texto. Esto significa que esos tres documentos tendrían 48 topics*
49	1.1.1.1.1.6	312375	1	Denoting, kanji
49	1.1.1.1.1.7	312588	2	Symbolic, circles

Tabla 16. Resultados del caso de estudio 5

Finalmente, en la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 31	1
32 - 39	2 (1 de los cuales no tiene coincidencia de palabras. Por tanto, ese documento se queda en el nivel 31)
40 - 45	3 (2 de los cuales no tiene coincidencia de palabras. Por tanto, esos documentos se quedan en el nivel 39)
46	5 (2 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 45)
47	25 (3 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 46)
48	159 (15 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 47)

49	596 (194 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 48)

Tabla 17. Resumen de los resultados del caso de estudio 5

5.1.2.2 Caso de estudio 6

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
1	0.5	2

Tabla 18. Hiperparámetros del caso de estudio 6

Los resultados obtenidos en este caso se resumen en la siguiente tabla donde se muestran únicamente algunas ramas del árbol de topics devuelto por el algoritmo.

Nivel	Rama	Topic	Número de documentos	Palabras más empleadas
0	0	0	1499	functional, algorithmic, learnt, setpoint, database, erties, vectorial, pointed, methodologies, problematic
1 - 42	0	1 - 3815	1499	Inquirí, unitary, neurally, learnt, weighted, systematic, recognizable, contextual, boundaries, controllable, static, optimality, frequent, modeled, characterised, objection, recognizable, features
43	1	3816	1133	relative, sufficiently, arising, operated, gilk, consequent, brownian, rationale, visited, turner
44	1.1	3817	1133	formulas, tively, ascertain, albrecht, accurately, frequently, alternatively, arithmetic, proposing, generatire
45	1.1.1	3818	474	decayed, setting, carrying, delusion, blackburn, lin, define, adjustable
46	1.1.1.1	3819	220	discussed, gest, fractal, lag, larly, branching, kirkpatrick, flatness, lowe, assimilation
47	1.1.1.1.1	3820	204	steepest, supply, bhalla, pattern, favored, generation, sinh, proce, agation, consecutively
48	1.1.1.1.1.1	3821	178	intention, talker, areas, johnson, houghton, obtaining, laminar, scanning, phasic, circuitry
49	1.1.1.1.1.1.1	3822	164	circuitry, chir, analogical, curried, outseg, voltages neuronal, implemented, systematic

49	1.1.1.1.1.1.2	53263	12	davies, michalski, Johnson, richardson, petersen, indexed, autistic, thompson, erkki, pause
49	1.1.1.1.1.1.3	410545	2	Theoretically, space, malignant

Tabla 19. Resultados del caso de estudio 6

Finalmente, en la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 42	1
43	2
44	4
45	11
46	36 (1 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 45)
47	114 (11 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 46)
48	297 (57 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 47)
49	648 (213 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 48)

Tabla 20. Resumen de los resultados del caso de estudio 6

5.1.2.3 Conclusiones tras la variación del hiperparámetro γ

Tras la realización de diversos experimentos, cambiando el valor del hiperparámetro γ, es decir los casos de estudio 2, 5 y 6, se pueden llegar a las siguientes conclusiones:

1) A mayor valor de γ, hay una menor dispersión del árbol de topics. Esto quiere decir que los temas son más genéricos y, por tanto, hay un mayor número de documentos que cumplen cierto criterio.
2) En el caso de una clasificación más selectiva, es decir, menores valores de γ, hay una mayor variedad de topics. Además, la dispersión del árbol comienza en niveles más cercanos al nodo raíz. Por ejemplo, en el caso de estudio 6, donde el valor de γ es 0.5, la diferencia entre documentos se aprecia en niveles más cercanos al raíz (en el nivel 43 ya se diferencian dos ramas del árbol) mientras que, para valores más altos, esta diferenciación se produce en niveles más bajos.

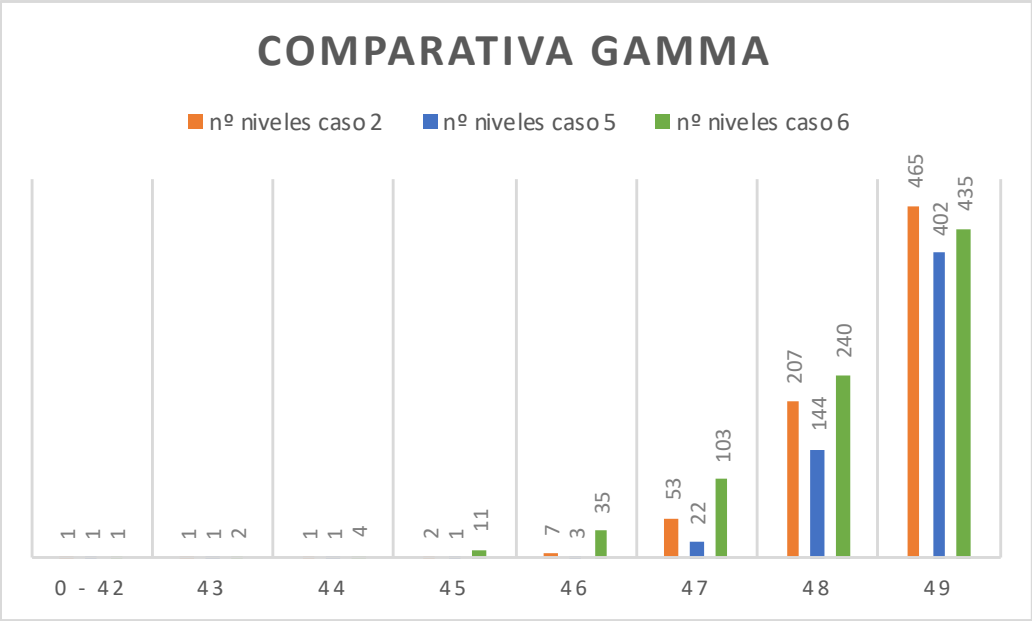

Figura 27. Gráfica comparativa de distintos valores del hiperparámetro γ

5.1.3 Análisis del hiperparámetro η

En este apartado se llevará a cabo un estudio del hiperparámetro η presente en el algoritmo implementado. En este caso, esta variable afecta al cálculo de pesos de cada nivel que, a su vez, se empleará en la asignación de un documento a determinado topic del árbol.

En los siguientes sub-apartados se presentan los diversos casos de estudio.

5.1.3.1 Caso de estudio 7

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
1	1	1

Tabla 21. Hiperparámetros del caso de estudio 7

Los resultados obtenidos en este caso se resumen en la siguiente tabla. Al igual que en el apartado anterior, se ha representado únicamente algunas ramas del árbol de topics devuelto por el algoritmo.

Nivel	Rama	Topic	Número de documentos	Palabras más empleadas
0	0	0	1499	signaled, noiseless, filterbank, frequent, chaos, spikes, informational, correlational, detector, trainable
1 - 43	0	379 - 27556	1499	Predictive, modeled, database, setpoint, regressive, validatory, testable, selective, motivate, directional, visualisation, systematic,

				objection, features, classifies, characterised, algorithmic
44	1	27557	1375	dominant, ocularity, mapped, eyes, developmental, orientational, eytan, competitive, borel
45	1.1	27558	974	patient, semantically, nance, attribute, damaged, optical, apical, farber, cleaned, damasio
46	1.1.1	27559	242	subsequently, sing, baxter, florham, biologically, ifr, pleasure, hecht, playing, motivated
47	1.1.1.1	27560	73	confusion, cell, absorb, error, extremely, jagota, close, achievement, picked, soybean
48	1.1.1.1.1	27561	37	establishing, enhances, pay, theo, inhomogeneous, sifter, rap, extremes, interactive, ild
49	1.1.1.1.1.1	27562	17	voronoi, eyes, veloped, header, rotational, gained, vestibule, reflexive, vet, store
49	1.1.1.1.1.2	62086	13	davies, michalski, Johnson, richardson, petersen, indexed, autistic, thompson, williamson, pause
49	1.1.1.1.1.3	436779	5	controlled
49	1.1.1.1.1.4	437281	2	Engineer, confer

Tabla 22. Resultados del caso de estudio 7

Finalmente, en la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 43	1
44	2
45	4 (1 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 44)
46	14 (1 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 45)
47	72 (11 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 46)
48	282 (58 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 47)
49	737 (279 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 48)

Tabla 23. Resumen de los resultados del caso de estudio 7

5.1.3.2 Caso de estudio 8

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
1	1	0.5

Tabla 24. Hiperparámetros del caso de estudio 8

Los resultados obtenidos en este caso se resumen en la siguiente tabla. Al igual que en el apartado anterior, se ha representado únicamente algunas ramas del árbol de topics devuelto por el algoritmo.

Nivel	Rama	Topic	Número de documentos	Palabras más empleadas
0	0	0	1499	noiseless, informational, codebook, coecient, bite, nolr, chaos, signaled, codeword, encompasses
1 - 45	0	1 - 5992	1499	Modeled, distributional, database, systematic, probable, parameterised, likely, neurally, resultant, testable, setpoint, characterised, recognizable, algorithmic, neuronal, memoryless, functional
46	1	5993	375	inseg, legal, behavioral, feel, enforce, occupant, fool, escapes, artificially, turned
47	1.1	5994	218	trivially, orthogonality, suitably, practically, jones, depen, eng, rotating, malignant, tectal
48	1.1.1	5995	178	Soil, pitches, syllables, faulty, neighboring, recruited, columbia, alto, proof
49	1.1.1.1	5996	141	controllable, modeled, controlling, robotic, learnt, tral, dynamical, motorola, arma, moves
49	1.1.1.2	61179	12	davies, michalski, Johnson, richardson, petersen, indexed, autistic, thompson, erkki, pause
49	1.1.1.2	174204	3	** En este caso ninguna palabra coincide con el texto. Esto significa que esos documentos tendrían 48 topics*
49	1.1.1.3	175425	4	Linkage, highlighted, spending
49	1.1.1.4	176245	5	Claiborne, unclear, includes
49	1.1.1.5	176750	6	dimen, stemming, softassign, boahen, acceptably, propulsion, continuously, attitude
49	1.1.1.6	176816	3	Replice, constrained, tapped, tendency

49	1.1.1.7	176901	2	Breast, linked, closed, distal
49	1.1.1.8	176909	2	Berg, Aston, localizer, condition, live

Tabla 25. Resultados del caso de estudio 8

Finalmente, en la *Tabla 26* se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 45	1
46	9
47	53 (4 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 46)
48	238 (61 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 47)
49	642 (219 de los cuales no tienen coincidencia de palabras. Por tanto, llegan al nivel 48)

Tabla 26. Resumen de los resultados del caso de estudio 8

5.1.3.3 Conclusiones tras la variación del hiperparámetro η

Tras la realización de diversos experimentos, cambiando el valor del hiperparámetro η, es decir los casos de estudio 2, 7 y 8, se pueden llegar a las siguientes conclusiones:

1) A mayor valor de η, hay una mayor dispersión del árbol de topics. Esto quiere decir que los temas son más específicos. Este crecimiento es más suave que en los casos de los hiperparámetros antes definidos. Sin embargo, hay un valor a partir del cual vuelve a decrecer esta dispersión ya que para valores constantes del resto de hiperparámetros, $\eta = 2$ tiene una menor dispersión que $\eta = 1$.

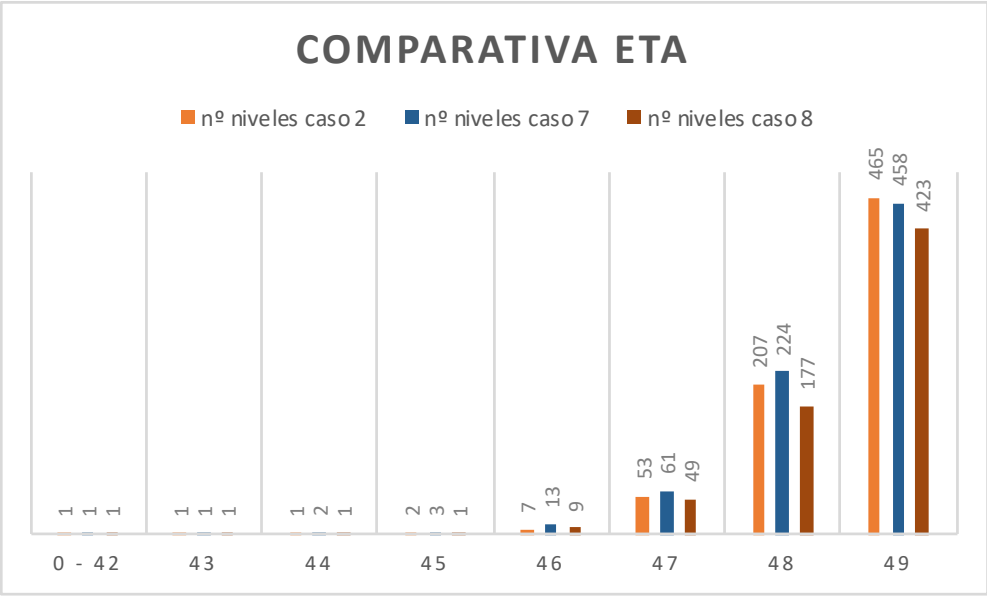

Figura 28. Gráfica comparativa de distintos valores del hiperparámetro η

5.1.4 Comparativa de hiperparámetros

En este apartado se presentan diversas gráficas que resumen la influencia de cada hiperparámetro en los resultados del algoritmo.

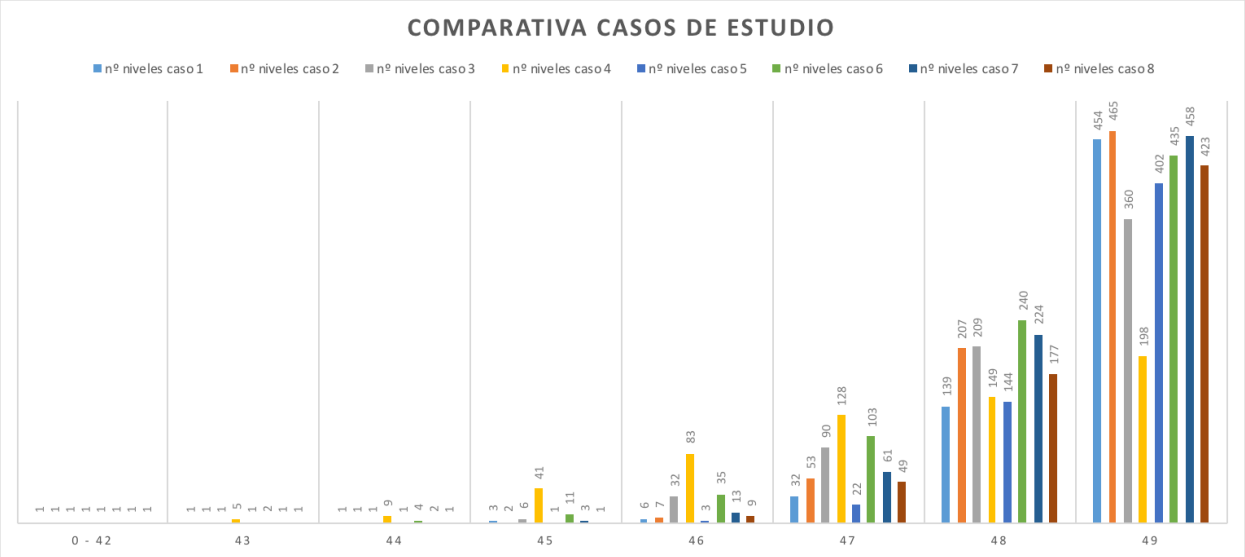

Figura 29. Comparativa general casos de estudio

Tras un análisis de los hiperparámetros aplicados a la base de datos NIPS podemos llegar a las siguientes conclusiones:

1) En el caso de estudio 4, se consigue una dispersión del árbol en niveles más cercanos al raíz que en el resto de casos. Además, el número de documentos que poseen coincidencia de topics en los últimos niveles es menor. Esto se traduce en que se ha logrado un análisis de topics más específicos en los que un menor número de documentos cumplen los requisitos.
2) En los casos de estudio 1 y 2, donde el hiperparámetro α es superior a 1, los documentos empiezan a divergir en niveles más altos, obteniendo un análisis más genérico. Esto ocurre también en los dos últimos casos de estudio, en los que la influencia de η es tan suave que no consigue superar al primer hiperparámetro.
3) También puede concluirse que el hiperparámetro α es el que más influencia tiene sobre los resultados, por tanto, en los casos de estudio en los que este valor permanece fijo, los resultados obtenidos son muy similares a pesar de variar el resto de hiperparámentros.

5.2 Base de datos CORA

En este apartado se analizará la segunda base de datos empleada en la validación del algoritmo propuesto en este trabajo. La base de datos posee un total de más de 11000 documentos de 80 categorías diferentes. En un primer análisis se han tomado 175 documentos del corpus y se ha modificado el valor del número de niveles para ver la influencia de este hiperparámetro. Los niveles que se han considerado han sido: 10, 20, 30 y 50.

Los siguientes subapartados se encuentran organizados en función del número de niveles.

5.2.1 Resultados con 10 niveles

Al igual que con la base de datos anterior se han modificado el resto de hiperparámetros, considerando los mismos casos de estudio.

5.2.1.1 Caso de estudio 1

En este primer caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
2	1	2

Tabla 27. CORA - Hiperparámetros del caso de estudio 1

Los resultados obtenidos en este caso se resumen en la siguiente tabla.

Nivel	Rama	Topic	Número de documentos	Palabras más empleadas
0	0	0	175	System, algorithm, paper, problem, model, method, data, result, program, using
1 - 5	0	33- 37	175	Artificial, achieved, presentation, elimination, virtual, dynamic, cluster, facilitate, decision, conclusion, Access, random, simultaneously, group, agreement, example
6	1	580	167	Responsive, situation, question, ccr, data, observed, approximation, speed, significant, logic
7	1.1	20671	41	Skincolor, perfect, world, people, uncontrolled, memory, smallest, sample, introduced, news
8	1.1.1	23262	5	Following, bitstring, often, rigorous, molecular, remains, building, hide
9	1.1.1.1	23836	2	Symbolically, channel
9	1.1.1.2	23937	2	Monitoring, shared, multicomputer
9	1.1.1.3	23962	1	shared
8	1.1.2	23533	9	Discovering, iterating, achieved, reaching, cryptographic, render, distance, local, vocabulary
9	1.1.2.1	23914	2	Patient, survival, eigenvalue, square, empirical, generate, curve
9	1.1.2.2	23950	4	Effectiveness, startup, dramatically, sheme, furthermore, multimodal, inductive
8	1.1.3	23878	16	Gain, visualize, effective, asymmetry, ensure, parameter, element, compressed, nature, oriented
9	1.1.3.1	23879	3	referent

9	1.1.3.2	23891	7	Recent, constraining, traditional, semantics, tractable, enough define, Stanford, remains
9	1.1.3.3	23911	3	Manner, focus, indexed, mechanism
9	1.1.3.4	23940	2	demonstrate
9	1.1.3.5	23959	1	Television, smoothing
8	1.1.4	23897	11	Sized, take, gray, targeted, reported, minimizing, circumscription, acknoledgement, alleviated, validate
9	1.1.4.1	23898	2	Paper, graysoftware, expressed, using, recursive, separate, multipass
9	1.1.4.2	23908	2	Unified, multirobot, serf, precision
9	1.1.4.3	23934	6	Estimating, character, recurrent, often, alternative, specificity, order, determining, prohibitive
9	1.1.4.4	23961	1	Priority

Tabla 28. CORA - Resultados del caso de estudio 1

Finalmente, en la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 5	1
6	2
7	4
8	22 (2 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 7)
9	75 (13 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 8)

Tabla 29. CORA - Resumen de los resultados del caso de estudio 1

La *Figura 30* representa algunas de las ramas del árbol de topics definido con estos parámetros. El código de colores de los niveles se resume en la siguiente tabla:

Color	Nivel
Celeste	0
Amarillo	1 - 3

Rojo	4
Verde	5
Rosa	6
Lila	7
Naranja	8
Añil	9

Tabla 30. CORA - Código de colores del árbol de topics

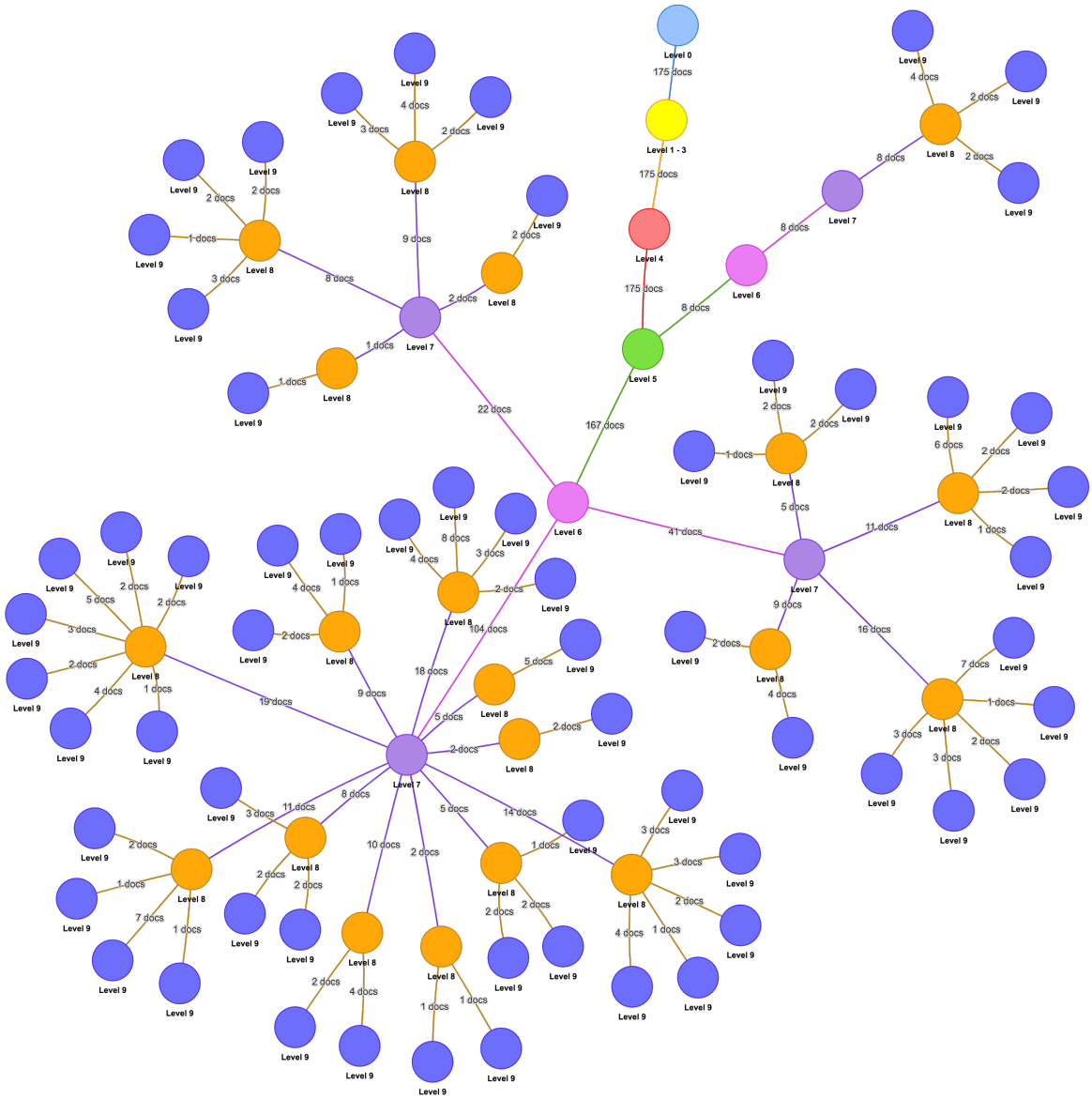

Figura 30. Árbol de topics resultante del caso de estudio 1 para la base de datos CORA

5.2.1.2 Caso de estudio 2

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
1	1	2

Tabla 31. CORA - Hiperparámetros del caso de estudio 2

Finalmente, en la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 3	1
4	2
5	2 (uno de los cuales no tiene coincidencia de palabras, por tanto, llegará hasta el nivel 4)
6	2 (uno de los cuales no tiene coincidencia de palabras, por tanto, llegará hasta el nivel 5)
7	7 (dos de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 6)
8	29 (siete de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 7)
9	82 (37 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 8)

Tabla 32. CORA - Resumen de los resultados del caso de estudio 2

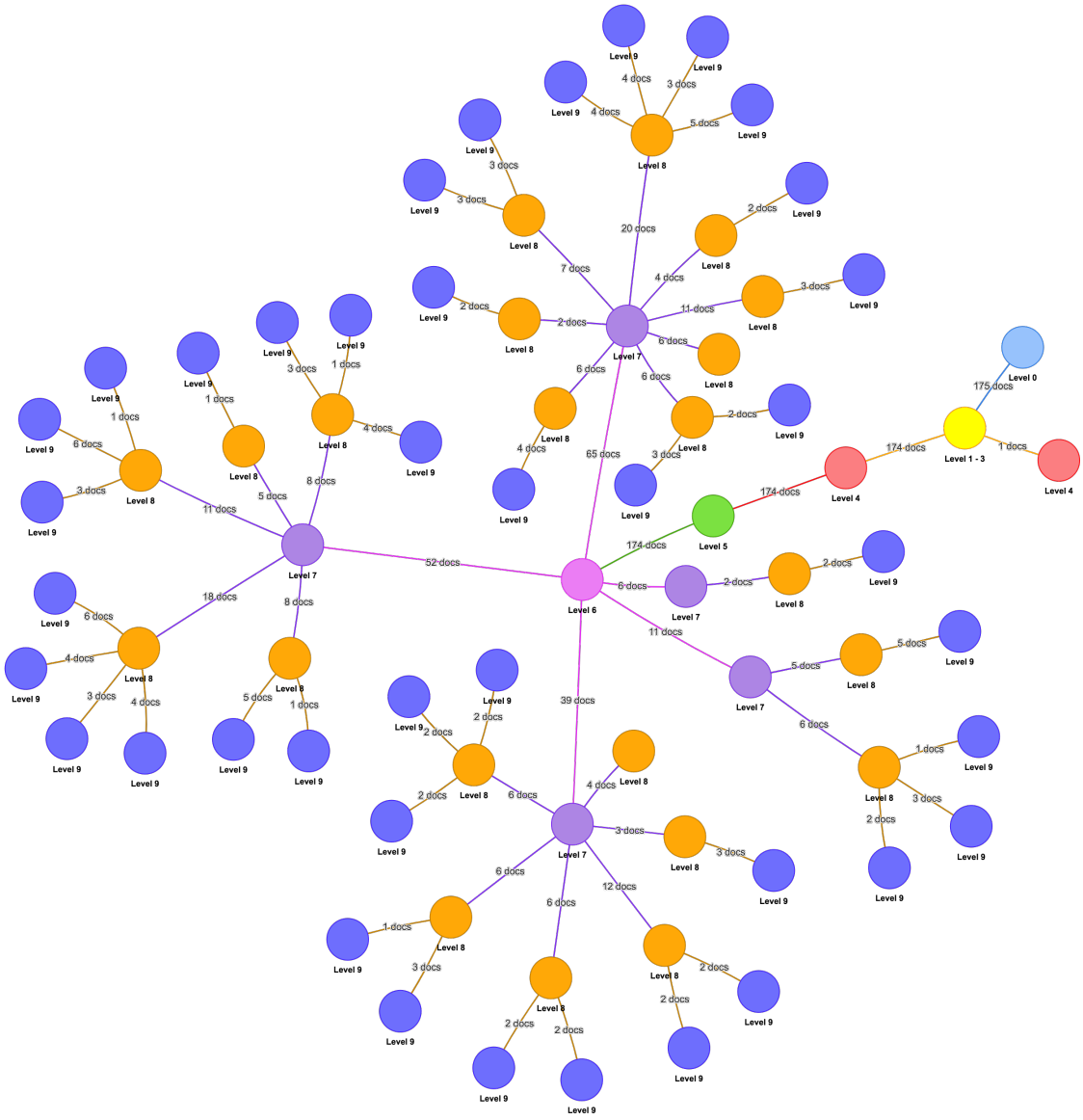

Figura 31. Árbol de topics de 10 niveles del caso de estudio 2 de la base de datos CORA

5.2.1.3 Caso de estudio 3

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
0.5	1	2

Tabla 33. CORA - Hiperparámetros del caso de estudio 3

En la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 3	1
4	2 (uno de los cuales no tiene coincidencia de palabras, por tanto, llegará hasta el nivel 3)
5	6 (cuatro de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 4)
6	15 (ocho de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 5)
7	35 (18 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 6)
8	65 (40 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 7)
9	114 (72 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 8)

Tabla 34. CORA - Resumen de los resultados del caso de estudio 3

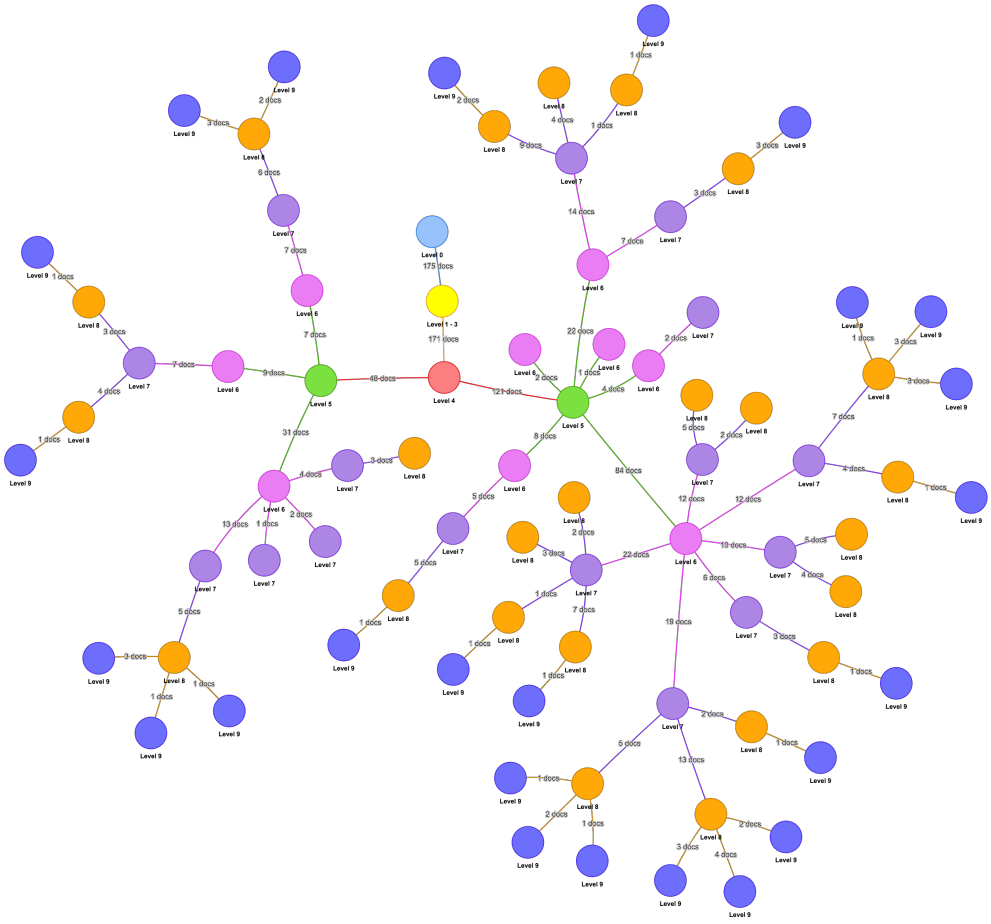

Figura 32. Árbol de topics de 10 niveles del caso de estudio 3 de la base de datos CORA

5.2.1.4 Caso de estudio 4

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
0.2	1	2

Tabla 35. CORA - Hiperparámetros del caso de estudio 4

En la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 2	1
3	3 (uno de los cuales no tiene coincidencia de palabras, por tanto, llegará hasta el nivel 2)
4	7 (tres de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 3)
5	19 (12 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 4)
6	37 (21 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 5)
7	57 (41 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 6)
8	89 (73 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 7)
9	134 (116 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 8)

Tabla 36. CORA - Resumen de los resultados del caso de estudio 4

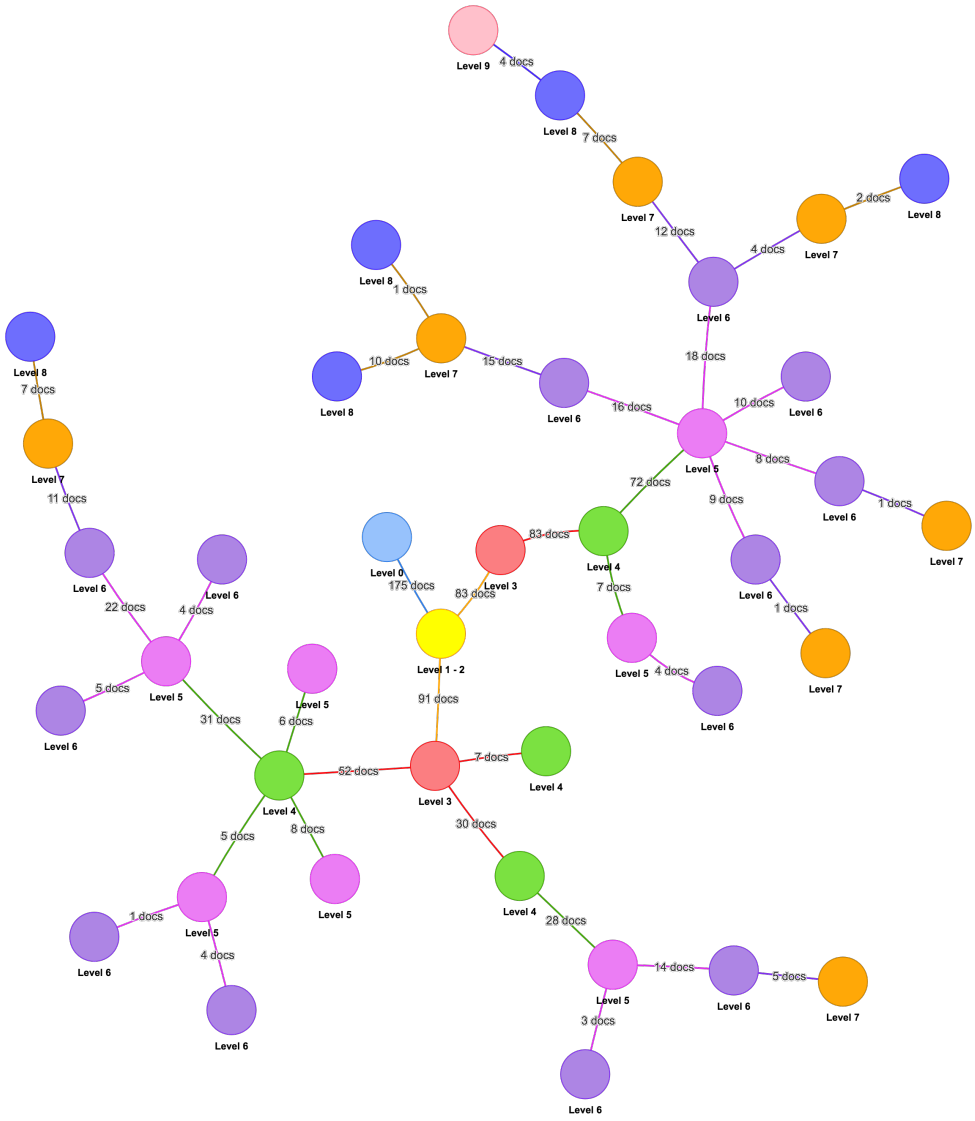

Figura 33. Árbol de topics de 10 niveles del caso de estudio 4 de la base de datos CORA

5.2.1.5 Caso de estudio 5

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
1	2	2

Tabla 37. CORA - Hiperparámetros del caso de estudio 5

En la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 4	1
5	2
6	3
7	10 (3 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 6)
8	29 (8 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 7)
9	81 (31 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 8)

Tabla 38. CORA - Resumen de los resultados del caso de estudio 5

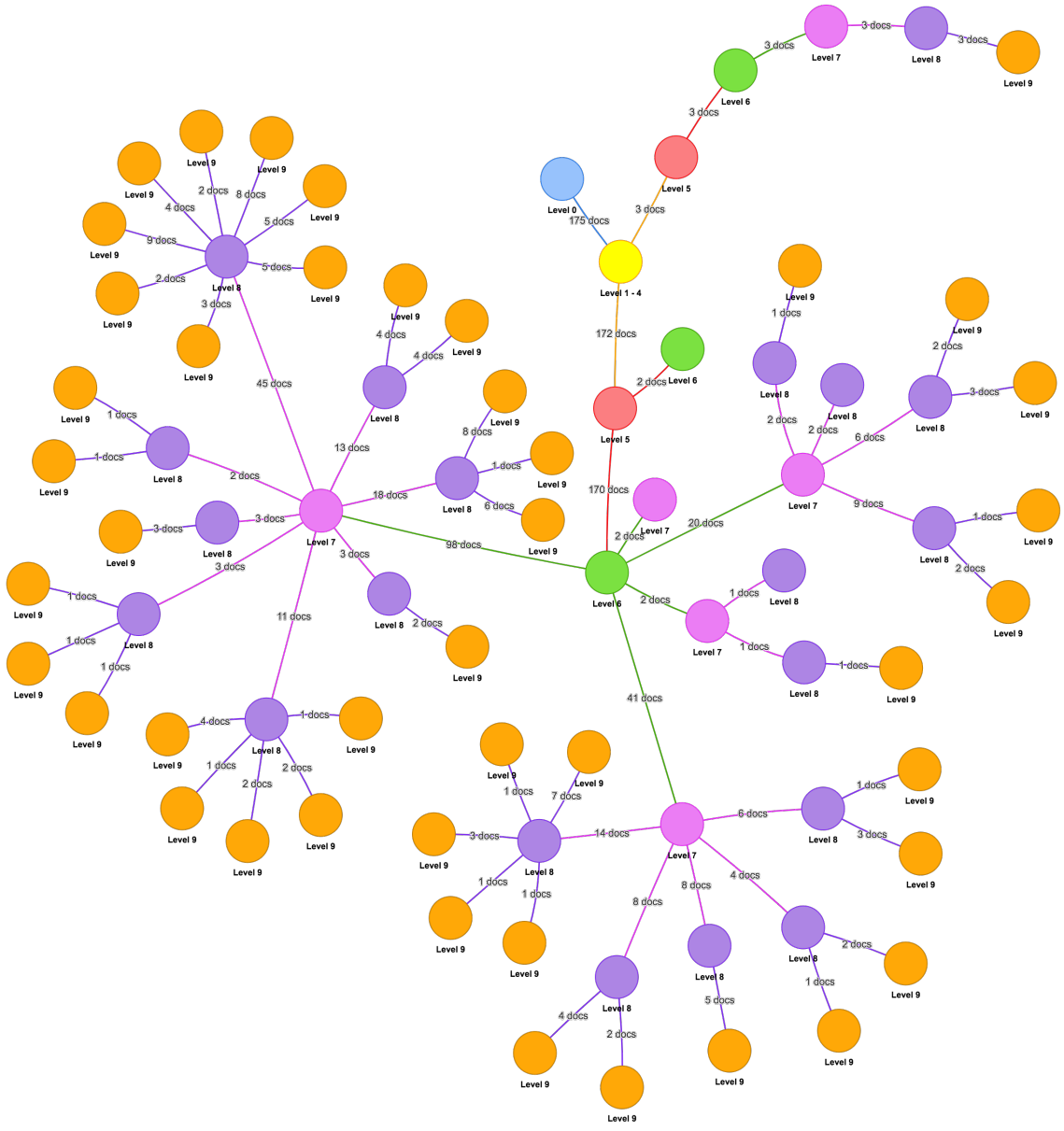

Figura 34. Árbol de topics de 10 niveles del caso de estudio 5 de la base de datos CORA

5.2.1.6 Caso de estudio 6

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
1	0.5	2

Tabla 39. CORA - Hiperparámetros del caso de estudio 6

En la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 4	1
5	2
6	5 (uno de los cuales no tiene coincidencia de palabras, por tanto, llegará hasta el nivel 5)
7	19 (3 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 6)
8	50 (11 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 7)
9	99 (46 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 8)

Tabla 40. CORA - Resumen de los resultados del caso de estudio 6

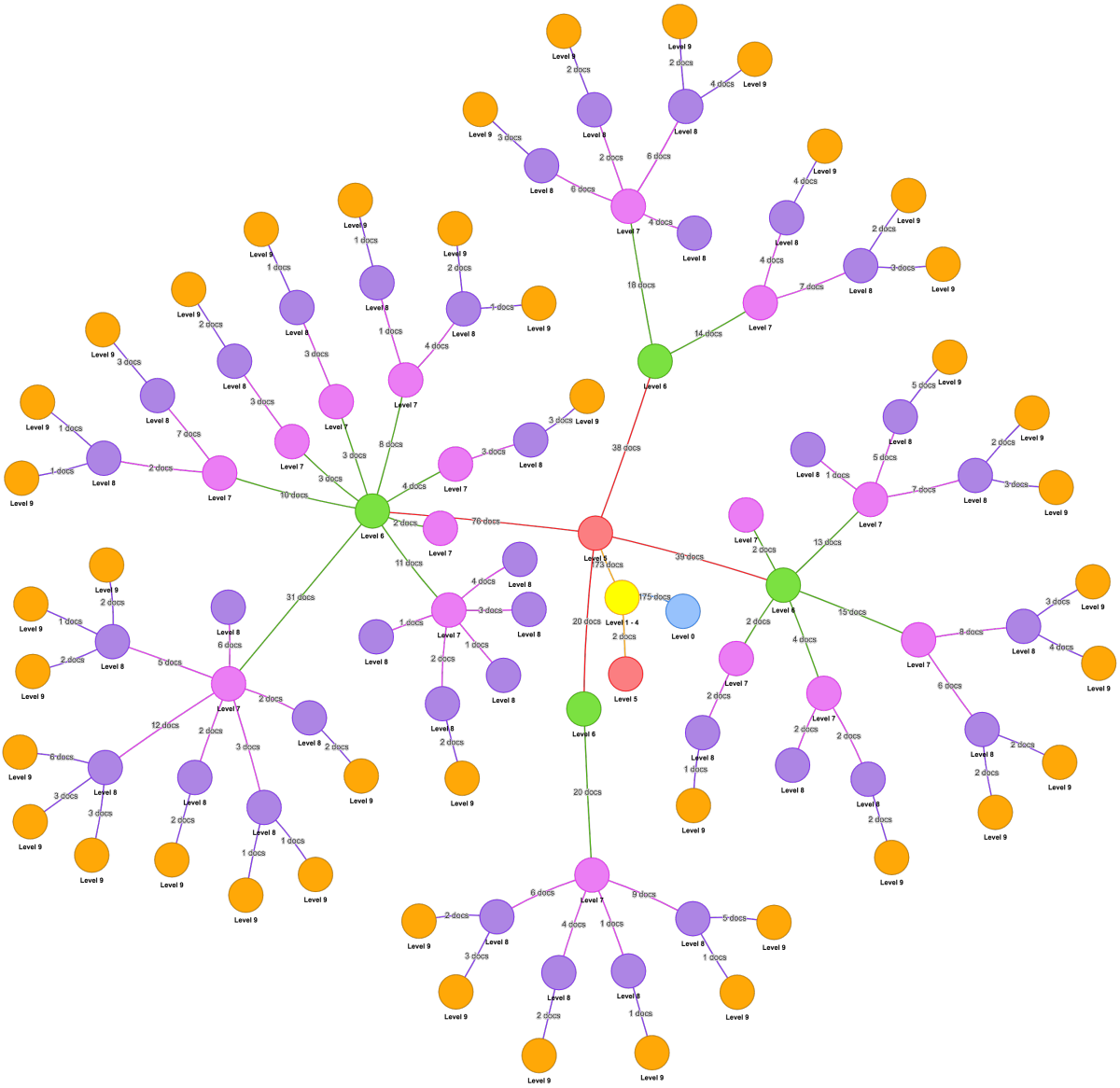

Figura 35. Árbol de topics de 10 niveles del caso de estudio 6 de la base de datos CORA

5.2.1.7 Caso de estudio 7

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
1	1	1

Tabla 41. CORA - Hiperparámetros del caso de estudio 7

En la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 5	1
6	5 (uno de los cuales no tiene coincidencia de palabras, por tanto, llegará hasta el nivel 5)
7	13 (3 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 6)
8	43 (15 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 7)
9	94 (36 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 8)

Tabla 42. CORA - Resumen de los resultados del caso de estudio 7

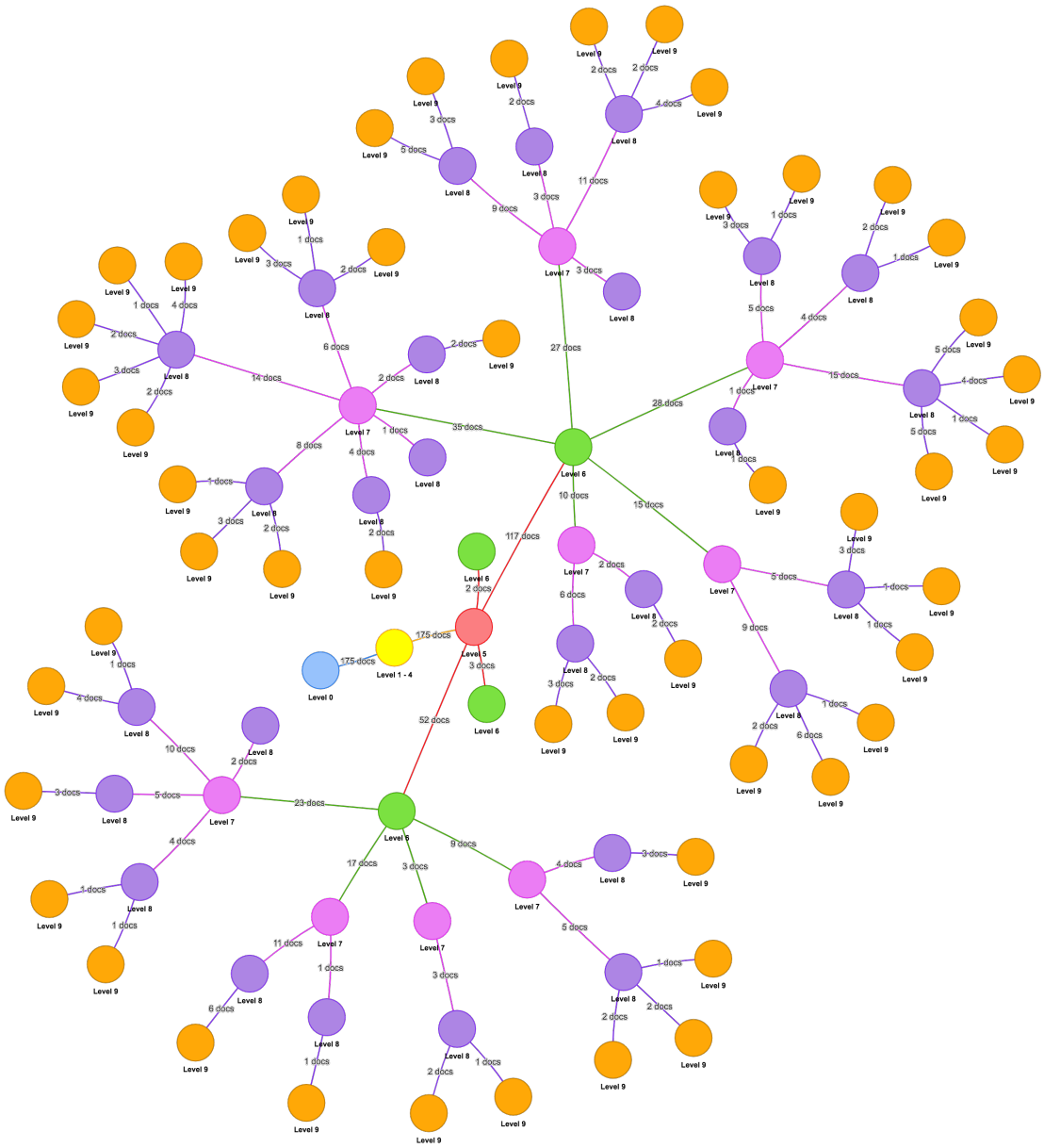

Figura 36. Árbol de topics de 10 niveles del caso de estudio 7 de la base de datos CORA

5.2.1.8 Caso de estudio 8

En este caso se analizarán los siguientes valores de los hiperparámetros:

α	γ	η
1	1	0.5

Tabla 43. CORA - Hiperparámetros del caso de estudio 8

En la siguiente tabla se muestran los resultados obtenidos en este caso de estudio a un nivel más genérico que nos permitirá hacer comparativas entre las diferentes ejecuciones del algoritmo.

Nivel	Número de niveles
0 - 4	1
5	2
6	3
7	11
8	36 (6 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 7)
9	90 (36 de los cuales no tienen coincidencia de palabras, por tanto, llegarán hasta el nivel 8)

Tabla 44. CORA - Resumen de los resultados del caso de estudio 8

Por último, se representa el árbol de topics resultante.

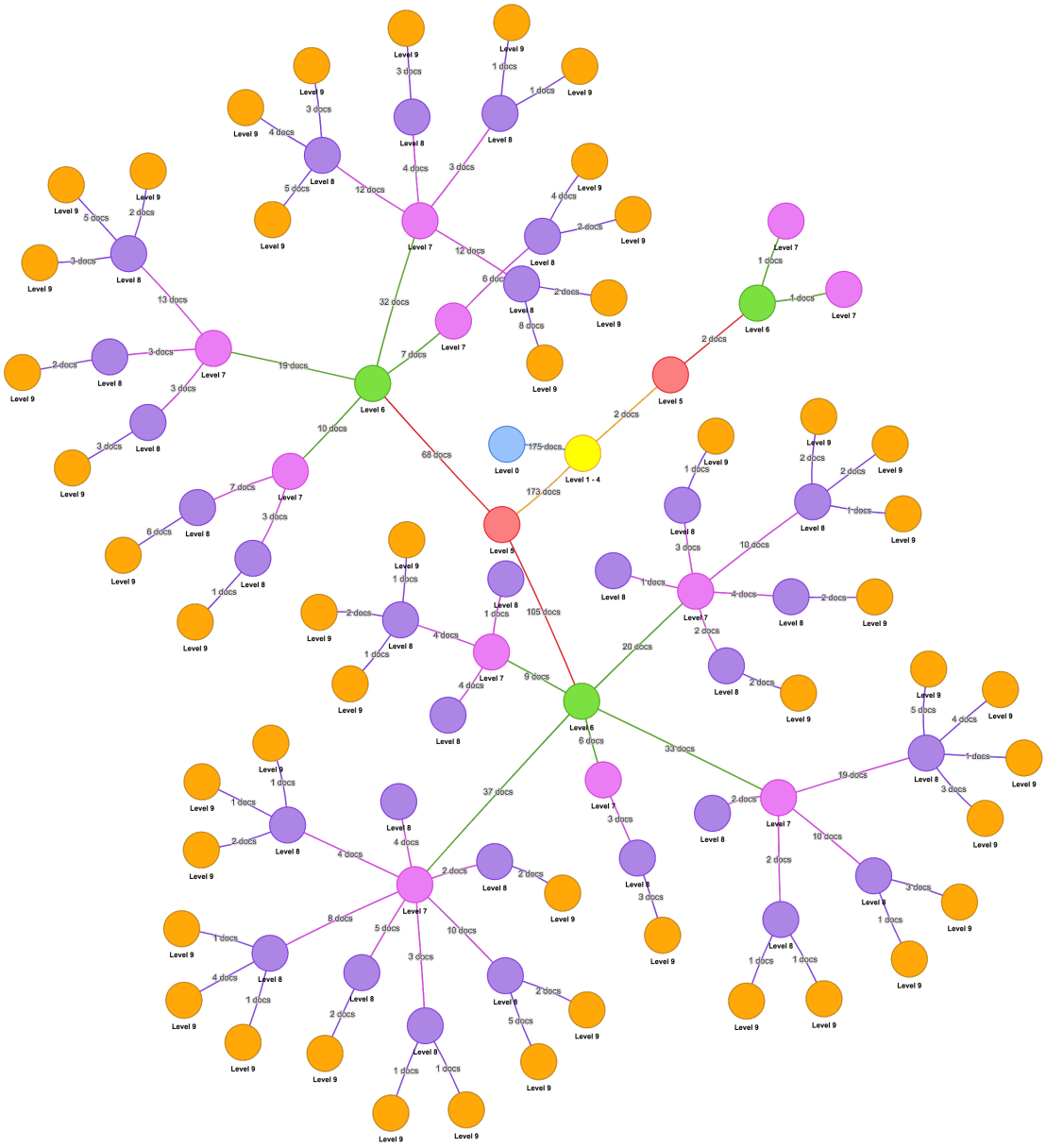

Figura 37. Árbol de topics de 10 niveles del caso de estudio 8 de la base de datos CORA

5.2.1.9 Comparativa casos de estudio

En este apartado se realizará una comparación de los resultados obtenidos en los diferentes casos de estudio y se extraerán algunas conclusiones. La *Figura 35* muestra una gráfica con el número de niveles para los distintos valores de los hiperparámetros.

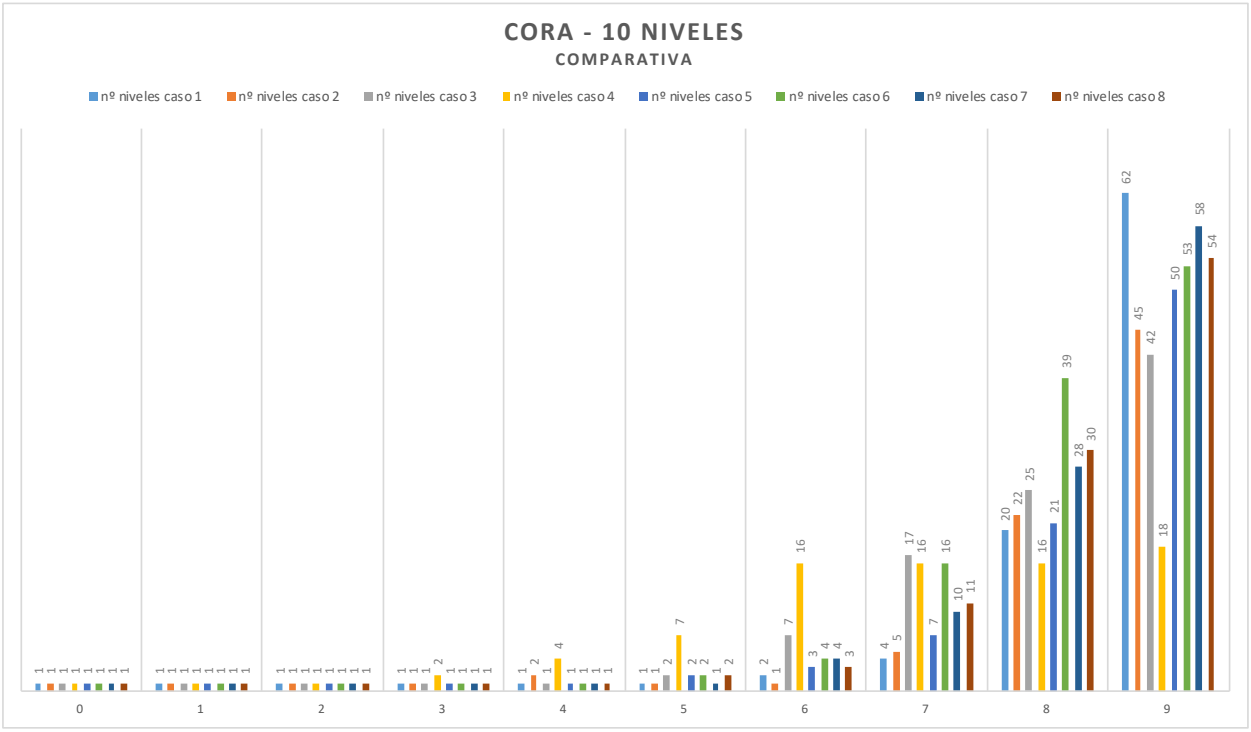

Figura 38. Gráfica comparativa de la base de datos CORA

Se puede apreciar que, en este caso, la estructura del árbol del topics es similar a la originada con la base de datos anterior. Por tanto, se concluye que el número de niveles no afecta a la dispersión del árbol, sino que, la adición de un mayor número de niveles trae consigo la generación de topics más generalistas. Esto quiere decir que la dispersión del árbol comenzará en niveles más alejados del nodo raíz cuando el número de niveles incrementa.

Al igual que con la base de datos anterior, se representa en la *Figura 39*, la comparativa de casos de estudio en función del hiperparámetro objeto de estudio.

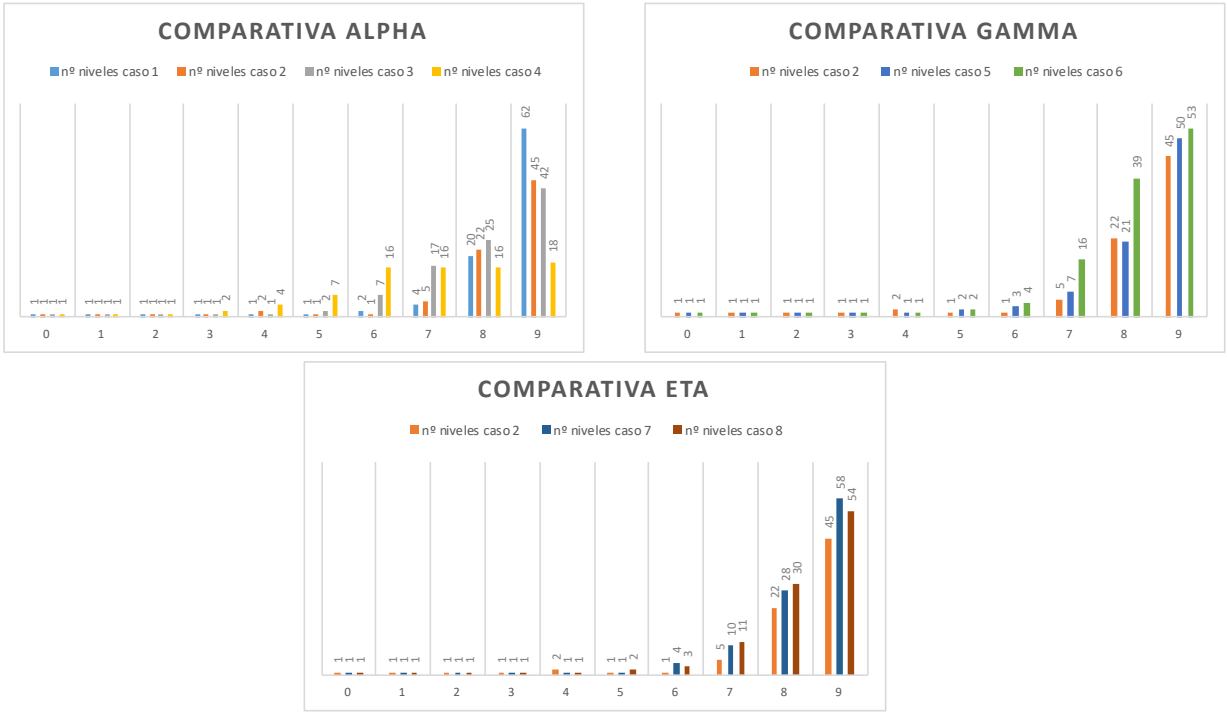

Figura 39. CORA - Comparativa parcial de cada hiperparámetro

Finalmente, se hace un estudio variando el número de niveles para la base de datos CORA. Los resultados se

presentan en el siguiente apartado.

5.2.2 Comparativa número de niveles

En este apartado se expondrán los resultados obtenidos modificando el número de niveles sobre el corpus de 175 documentos de la base de datos CORA.

En las siguientes gráficas se mostrarán el conjunto de niveles/topics tanto con coincidencia de documentos como sin ella para cada uno de los experimentos.

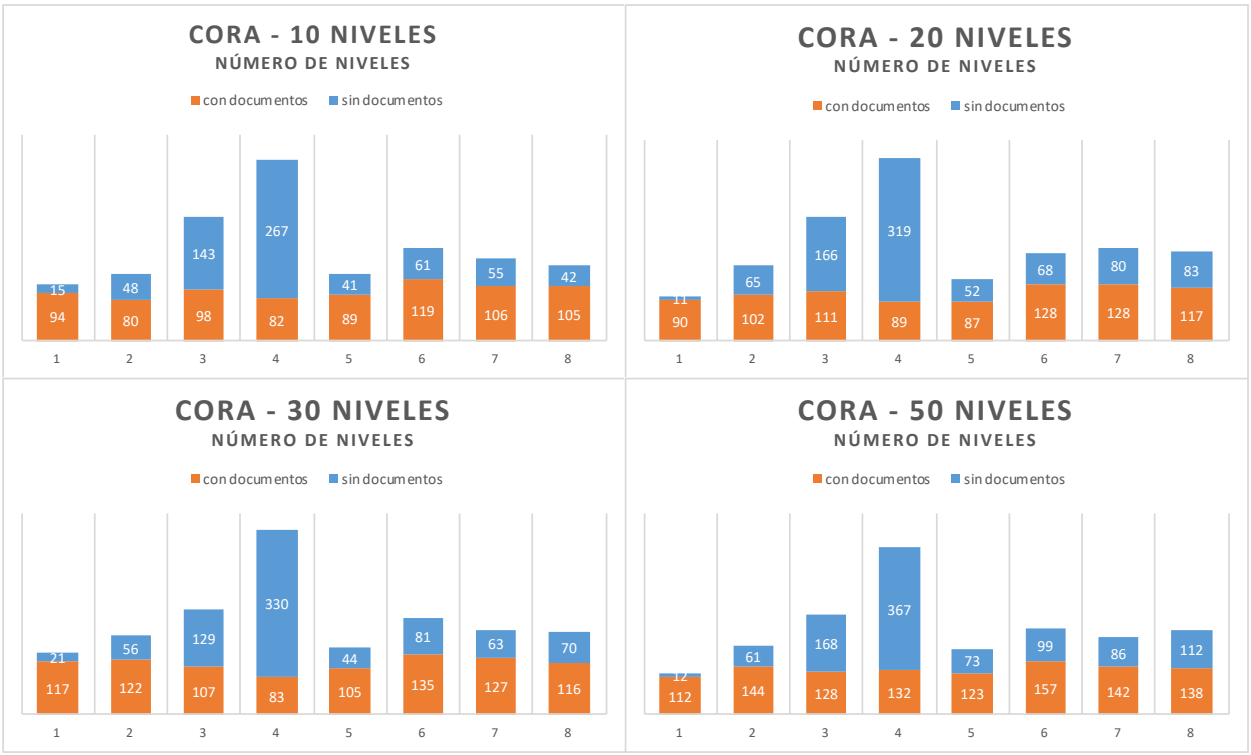

Figura 40. Representación de topics en función del número de niveles y del experimento

En la *Figura 40*, se pueden apreciar cuatro gráficas, cada una de las cuales hace referencia a los experimentos realizados fijando el hiperparámetro *número de niveles* a determinado valor. Se puede observar cómo los resultados son similares en los cuatro casos. Por tanto, se puede concluir que el número de niveles fijado no es determinante para los resultados del algoritmo. *Esto quiere decir que la forma del árbol resultante en cada experimento es en cierta medida independiente del número de niveles*. Por ejemplo, en un experimento de 10 niveles con el resto de hiperparámetro fijos, el árbol comenzará a divergir en el nivel 5 mientras que, en otro experimento, con los mismos valores de hiperparámetros salvo el número de niveles que se fijará en 50, el árbol comenzará a divergir en el nivel 45. Este hecho puede apreciarse en la *Figura 41*.

Por otro lado, puede observarse cómo en el caso de estudio 4 el número de topics totales es muy alto, es decir, es un estudio más específico. Sin embargo, es este carácter específico el que hace que muy pocos documentos cumplan los requisitos de pertenencia a un topic y, por tanto, hay un gran número de topics que no tienen asignados documentos. Se puede ver cómo este número incrementa a medida que el hiperparámetro α disminuye.

Otra característica ya mencionada en apartados anteriores es la influencia de cada hiperparámentro, se puede observar cómo una variación del valor de α supone una mayor diferenciación de los resultados que un cambio en el resto de hiperparámetros cuyos efectos son menores.

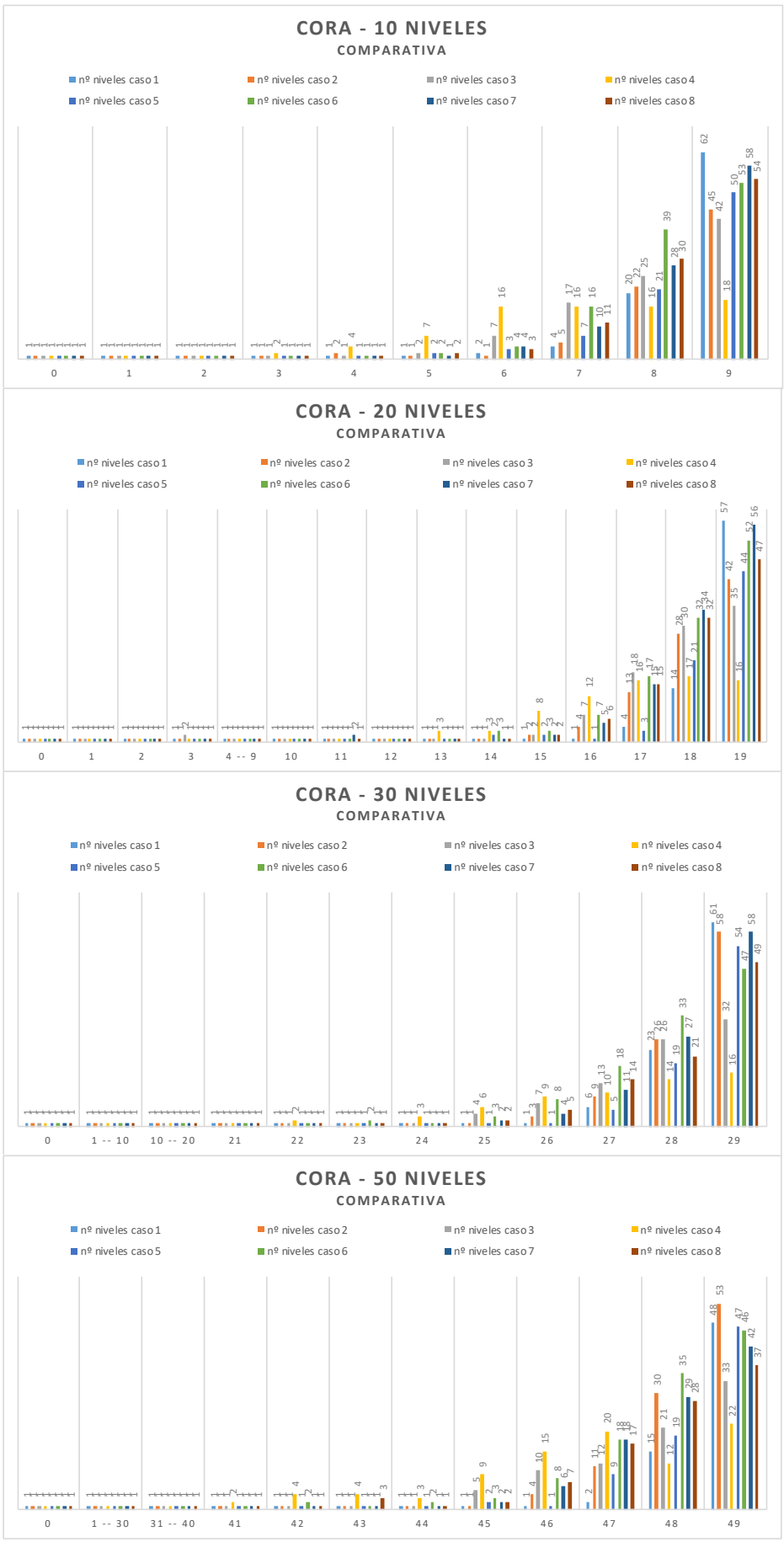

Figura 41. CORA - Representación de topics resultante de cada experimento

6 CONCLUSIONES

The true sign of intelligence is not knowledge but imagination

- Albert Einstein -

Este proyecto tiene como objetivo profundizar en uno de los campos de aplicación de machine learning. En concreto, se ha implementado un algoritmo de clustering de documentos para solventar el problema de topic modeling. El lenguaje elegido para dicha implementación ha sido Python debido a la importancia que está alcanzando en los últimos años gracias a la gran cantidad de librerías disponibles, permitiendo una implementación más sencilla que con otros tipos de lenguajes. En primer lugar, se ha realizado una investigación de los modelos presentes en el estado del arte de este campo, empezando por el más simple, LDA, que ha servido de punto de partida para el resto de algoritmos, y terminando por algoritmos más avanzados en los que se tienen en cuenta, no sólo el contenido del documento, sino también el autor o la fecha de publicación que sirven para establecer otro tipo de relaciones entre documentos. En este proyecto, se ha optado por el método hLDA propuesto por David M. Blei (Blei, Griffiths, Jordan, & Tenebaum, Hierarchical Topic Models and the Nested Chinese Restaurant Process, 2013) en el que haciendo uso del algoritmo nCRP y la distribución de Dirichlet consigue clasificar los documentos en un árbol jerárquico de topics. El método de nCRP se emplea para la construcción del árbol de topics, mientras que la distribución de Dirichlet se usa para la asignación de un documento a determinado topic. Por último, se emplea como función a optimizar la verosimilitud, es decir, se pretende minimizar el error cometido. Este cálculo es muy complejo y, en su lugar, se emplea una aproximación haciendo uso del proceso de muestreo de Gibbs.

Finalmente, para validar el algoritmo se han empleado dos bases de datos diferentes: NIPS y CORA. La primera de ellas compuesta por 1500 papers sobre sistemas de procesado de información neuronal y la segunda por más de 11000 papers de 80 clases diferentes.

Se han evaluado la influencia de los cuatro hiperparámetros presentes en el algoritmo: número de niveles, α, γ y η y se extraen las siguientes conclusiones:

1) El número de niveles no tiene una gran influencia en la topología del árbol de topics. Añadir más niveles únicamente hace que incremente en número de topics genéricos cuyos criterios de pertenencia son cumplidos por todos los documentos del corpus.
2) Un mayor valor de α hace que el estudio realizado sea más genérico, extrayendo temas comunes a un gran número de documentos, mientras que con valores más pequeños se consiguen árboles de topics más específicos. Sin embargo, se ha de considerar que bajar mucho el valor del hiperparámetro hace que el estudio sea demasiado específico y, por tanto, pocos documentos cumplen los requisitos de pertenencia a los topics de niveles más lejanos al nodo raíz. Esto puede verse en el caso de estudio 4 del capítulo anterior.
3) El mismo efecto que el anterior tiene la modificación del hiperparámetro γ. Sin embargo, la influencia es más suave y hay un mayor número de documentos que cumplen los requisitos de pertenencia en los nodos más alejados del raíz.
4) El caso del hiperparámetro η es diferente a los anteriores. La dispersión del árbol de topics comienza a crecer con η hasta que llega a un valor a partir del cual vuelve a decrecer.

Por último, comentar que la elección de estos hiperparámetros será muy importante para obtener unos resultados adecuados en función del estudio que se quiera realizar.

REFERENCIAS

(s.f.). Obtenido de GitHub: https://github.com/dongwookim-ml/python-topic-model

Aggarwal, C., & Zhai, C. (2012). *Text Mining Data.* Springer.

Aldous, D. (1985). Exchangeability and related topics. *École d'été de probabilités de Saint-Flour, XIII -1983*, 1-198.

Alghamdi, R., & Alfalqi, K. (2015). A Survey of Topic Modeling in Text Mining. *International Journal of Advanced Computer Science and Applications, 6*(1), 147-153.

Alpaydin, E. (2014). *Introduction to Machine Learning.* MIT Press.

Andrew McCallum, U. o. (s.f.). Obtenido de https://people.cs.umass.edu/~mccallum/data.html

Blei Lab. (s.f.). Obtenido de https://github.com/blei-lab/edward

Blei, D., & Lafferty, J. (2006). Dynamic Topic Models. *Proceedings of the 23rd international conference on Machine learning (ICML '06)*, 113-120.

Blei, D., & Lafferty, J. (2007). A Correlated Topic Model of Science. *The Annals of Applied Statistics, 1*(1), 17-35.

Blei, D., & Lafferty, J. (2009). Topic Models.

Blei, D., & McAuliffe, J. (2007). Supervised Topic Models. *Proceedings of the 20th International Conference on Neural Information Processing Systems*.

Blei, D., & Moreno, P. (2001). Topic segmentation with an aspect hidden Markov model. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 343-348.

Blei, D., Griffiths, T., & Jordan, M. (2010). The Nested Chinese Restaurant Process and Bayesian Nonparametric Inference of Topic Hierarchies. *Journal of the ACM, 57*(2).

Blei, D., Griffiths, T., Jordan, M., & Tenebaum, J. (2013). Hierarchical Topic Models and the Nested Chinese Restaurant Process.

Center of Machine Learning and Intelligent Systems, University of California. (s.f.). *UCI Machine Learning Repository*. Obtenido de https://archive.ics.uci.edu/ml/datasets/bag+of+words

D. M. Blei, A. Y. (2003). Latent Dirichlet Allocation. *The Journal of Machine Learning Research, 3*, 99-1022.

Frigyik, B. A., Kapila, A., & Gupta, M. R. (2010). *Introduction to the Dirichlet Distribution and Related Processes.* University of Washington, Department of Electrical Engineering, Seattle.

Gershman, S., & Blei, D. (2011). A tutorial on Bayesian nonparametric models. *Journal of Mathematical Psychology*.

Griffiths, T., & Steyvers, M. (2004). Finding scientifics topics. *Proceedings of the National academy of Sciences of the United States of America 101*, 5228-5235.

Hofmann, T. (1999). Probabilistic latent semantic indexing . *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 50-57.

https://github.com/dongwookim-ml/python-topic-model/blob/master/ptm/utils.py. (s.f.).

Li, W., & McCallum, A. (2006). Pachinko Allocation: DAG-structured mixture models of topic correlations. *Proceedings of the 23rd international conference on Machine Learning*.

Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. *University of California, School of Information and Computer Science*. Irvine, CA.

Mimno, D., & McCallum, A. (2008). Topic models conditioned on arbitrary features with Dirichlet-multinomial regression. *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, 411-418.

Mohit, B. (2014). Named Entity Recognition. En I. Zitouni, *Natural Language Processing of Semitic Languages* (págs. 221-245). Springer Berlin Heidelberg.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective.* The MIT Press.

Nishma Laitonjam, V. P. (2015). Topic Modelling for Songs. *IEEE 2015 International Conference on Information Technology*.

Rosen-Zvi, M., Griffiths, T., Steyvers, M., & Smyth, P. (2004). The Authos-Topic Model for Authors and Documents. *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, 487-494.

V. Jelisavčić, B. F. (2012). Topic Model and Advanced Algorithms for Profiling of Knowledge in Scientific Papers. *MIPRO, 2012 Proceedings of the 35th International Convention*, 1030-1035.

Wang, X., McCallum, A., & Wei, X. (2007). Topical n-grams: Phrase and topic discovery, with an application to information retrieval. *Proceeding of the 7th IEEE International Conference on Data Mining*, 697-702.

Wiering, M. y. (2012). *Reinforcement learning: State-of-the-art.* Springer.

ANEXO


Level 0. Words: recurrently, stringent, symbolic, sequencing, gramatical, finitely, sequencer, static, translate, languages
Level 1 - 37. Words: Circuitry, analogical, modeled, database, distributional, functional, methologies, classifies, learnt, algorithmic
Level 38 - 44. Words: Speechreading, recognizable, systematic, speaking, visualisation, imaginary, motivate, connectivity, representational, directional
Level 45. Words: Cost, accent, OSPF, ZFE, efficient, run, due, schemes
Level 46. Words: Explanatory, promote, cyclic, inserted, sand, viewed, readily, encouraging
Level 46. Words: chaining, succession, stork, spec, vision, setting, finance, static, tractor, conserved
Level 46. Words: beaverton, blocking, sajda, geometrical, correspon, children, permitted, iter, baum, lose
Level 47. Words: nikhil, invertebrates, substituted, acous, internet, stronger, learnability, ttk, intervention, drain
Level 48. Words: label, fitting, demonstrating, guaranteeing, decimation, discriminative, reduced, nist, selected, beneficial
Level 49. Words: raising, probabili
Level 49. Words: physiologically, notoriously
Level 49. Words: focused, rig
Level 49. Words: partly
Level 49. Words: truncating
Level 49. Words: founda
Level 49. Words: served, blue, printing, studies, rhythmic, stewart
Level 48. Words: orderly, discriminates, latent, correlated, piepenbrock, lstm, independently, snr, respec, sufficient
Level 49. Words: filtering, pauern
Level 49. Words: ylz, markovjan, resetting, choices, coelho, clw, founda, environmental, drifting, conjunctive
Level 49. Words: gullapalli, academy, simard, spaces, neuron, rodney
Level 49. Words: intelli, stably
Level 48. Words: acm, char, chosen, helping, wyatt, regularization, velocity, miller, keyboard, billion
Level 49. Words: kitten, otto, hash
Level 49. Words: biases, autoassociator, impression, connectionism
Level 49. Words: categorize, paradox, motivates, validate, klein, relating, afford, statist, jection, schuster
Level 49. Words: silenced, adapt, physically, assiciates, obeying, bond
Level 49. Words: mainen
Level 49. Words: equipartition, determines, interactive, mezard, invert
Level 49. Words: candidates
Level 49. Words: linsker, batches, krug, filterbank, maxima, maximal, adjectives, snr, practically
Level 49. Words: chapel, observables
Level 49. Words: server, suggestive, begun, ein, deconvolution, detected
Level 49. Words: quantisation, huang, abundances, mother
Level 48. Words: partic, summarized, menzel, strike, backpropagate, genera, ime, spoken, servation, settling
Level 49. Words: andersen
Level 49. Words: depending, nonnegative, suggestion, drawn, structuring, substructures, june, virtues, mapped, fido
Level 48. Words: yoshua, owen, digitized, hildreth, online, objec, acceptable, interac, harmonies, resolution
Level 49. Words: passage, suf
Level 49. Words: bible
Level 49. Words: iri, adjectives, cnv, submodule, cognitive, revenue, coisr, saying, inhibiting, equalizer
Level 48. Words: favored, impractical, performance, broadcast, believe, arises, kyoto, georg, resistive, kairiss
Level 49. Words: decreasing, mcnaughton, mouth, behavioral, frozen
Level 49. Words: bauer
Level 49. Words: ipsp
Level 49. Words: vector, motivates, rapid, diffuse, seminal, produced, schulten, persistence, denoting, subsequently
Level 48. Words: marked, continuously, named, lithography, assertion, conceptually, founded, equalizer, ariaboost, baram
Level 49. Words: iden, bed, serially, nato, codebook, maintaining, traditionally, egelhaaf, referenced, simplistic
Level 48. Words: algorit, mag, discrim, bridle, highlight, ward, eventually, valley, region
Level 49. Words: diagonalization, resource
Level 49. Words: viewing, info, skew, experienced, ford, movshon, facial, suppresses, donald, suited
Level 49. Words: standardized, subpart, duration, leen, divided, nakahara, interesting, pitiform, cohort
Level 49. Words: unrepresented, elsevier, filter, til, tried, enforcing
Level 49. Words: agrin
Level 49. Words: plotted, bout, instrument, science, gainesville, indication, alzheimer
Level 48. Words: ild, tangent, def, traces, distances, agreed, almeida, bowl, asa, direct
Level 49. Words: ntn, layer, tokyo, msec, tangent, obstacles, micron, fully
Level 47. Words: caller, gen, sing, enforcing, received, restart, kamimura, van, satisfy, fined
Level 48. Words: basilar, stimulus, recipe, student, mar, resemblance, remarkably, west, summit, complexity
Level 47. Words: learner, investigation, adversary, selection, segregate, recon, classifier, meta, segmented, recognizer
Level 48. Words: severly, judd
Level 48. Words: morning, concentrated, plexiform, sharing, div, summer, iter, yamashita, written, juang
Level 49. Words: styles, capac, mle, dissertation
Level 49. Words: perfusion, agc, edge_x, screening
Level 49. Words: saccade, zoo, fatigue, matsuoka, topography, structured, harrison, ryckebusch, stiffness, imitating
Level 49. Words: say, tones
Level 49. Words: occasion, behavioural, utilization, mume, finer, hot, mislabeled
Level 49. Words: err
Level 48. Words: cis, cease, straint, belief, possi, shatter, cnap, environment, cleereman, leary
Level 49. Words: robot, centering, predic, perceptually, airborne, lagged
Level 49. Words: linearities, alm, heavy, pbil, screen
Level 49. Words: maintain, frame, rock, horseshoe, induction, external, asked, difficulties, convenience, government
Level 49. Words: wish, share, overlaid, dense, transconductance, exhibiting, factorial, possessing, workshop
Level 49. Words: multiply, car, automatically, exponentiated, relates, regularization, obtain, deter
Level 49. Words: dejong, leftmost
Level 48. Words: leveraging, bpp, grouped, calculated, noted, sphering, concurrent, prop
Level 49. Words: endmember, infor, cue, algorit, invertebrate
Level 49. Words: williamson, lakoff, chua, ances, retention, eeckman, correspondingly, fisher, geometric, exam
Level 49. Words: advanced, perceptron, reinforcement, overlapped, embed
Level 49. Words: favour, sigmoidal, creates, sciences, ampa, buffering
Level 49. Words: thereof
Level 49. Words: unresolved, stb, raster, uncertain, leonard, bpq
Level 49. Words: georg, gathered, guided, faithful, franco
Level 48. Words: utile, edition, textured, tuned, tunable, interesting, closest, illusory, contrast, ker
Level 49. Words: organize, functioning, hancock, select, dejong, capac, lap, quantizing, wiegerinck, redun
Level 49. Words: memorandum, tracing, exactly, closes, couple
Level 49. Words: goodhill, ceases, alterna, option, artmap, referenced, presented, classi, suited, equipped
Level 49. Words: Davies, michalski, johnson, richardson, petersen, indexed, autistic, thompson, erkki, williamson
Level 49. Words: Smoothing
Level 49. Words: Postdoctoral, subunit
Level 47. Words: Scaled, persist, bumptrees, altogether, static, guaranteed, vanishing
Level 48. Words: Goodness, robertson, cooling, leonard, dash, gener, render
Level 48. Words: dempster, able, supplying, architecture, sphering, applicability, adjust, alent, ulated
Level 49. Words: unclamped, principles, africa
Level 49. Words: enhancing
Level 49. Words: uniquely, adjust, force
Level 47. Words: smoothed, exert, vergence, resistances, staggered, preserving, contributed, reconstructing, integral
Level 48. Words: smoothed, attias, consequences, comparator, provement, bending, compatibility, derlying, holistic
Level 48. Words: nadal, deliver, intrinsically, mistaken, knudsen, goudreau, ternary, experimenting, singh, stating
Level 48. Words: batches
Level 48. Words: gabriel, topographically, concurrently, identify, gpc
Level 49. Words: deeper, attias
Level 49. Words: voltages, passenger, asynchronously
Level 49. Words: geometrically, terence, schematically
Level 49. Words: netw
Level 48. Words: infeasible, lemasson, accuracy, mother, primates, actuator, relearning, sparseness, peaked, implicated
Level 49. Words: realized, break, pertone, danish, essence
Level 49. Words: asso, cholesterol, knudsen, ambiguous, spond
Level 49. Words: erkki, loaded, france, ate, pervasive, revealing, key, eventually, strengthening, accounting
Level 49. Words: wished
Level 49. Words: dom, hallucination, reusable, connecting, role, hawkin, requiring, goudreau
Level 49. Words: monitored, subregion, introduction, settle, parentheses, evolved, temporally, lowe
Level 49. Words: morph, equalities, cone
Level 49. Words: outcomes, fraction, appoximating, surrounded
Level 49. Words: delayed, tinit, ruyter
Level 49. Words: sigmoid, university, exploitation, leary, bulsara
Level 49. Words: nowlan, siegel, pentium, nishomori, locative, expresses, shankel, organ, nonterminal, redun
Level 49. Words: repeatable
Level 48. Words: instantiation, mitchell, tives, quenched, sigir, mirrored, randomly, tracked
Level 49. Words: superiority, noordewier, hermissenda
Level 49. Words: stemming, email, drf, roles, efron, frames, hammer, istic, developed, implementable
Level 49. Words: polar, transmitter, breakdown, simulation, aligned, angles, power, diode, string, modify
Level 49. Words: subutai, battery, driving, vanishing, fine, facing, overload, stably, intimately, attend
Level 49. Words: derivative, exten
Level 49. Words: bmu, proves, signif, grinvald, wsi, detrimental, nec, slot, reso
Level 49. Words: accounting
Level 49. Words: kawamoto, reproducing, focuses, significantly, schwarze, generaliza
Level 49. Words: institute, neuromodulation, neocognitron, surrounding, logothetis, rieke, learnt, divergences, lighter, constructive
Level 49. Words: played, seika
Level 49. Words: learnt
Level 49. Words: dealing, sufficiency, decides, recover, biologically, exhibit, perturbation, skewness, unbiasedness
Level 49. Words: skeletonisation, reproducing, science, smoothing
Level 49. Words: gence, insertion, barely, average, postulated, erroneous, acquiring
Level 49. Words: perfor, iri, incrementing, irrespective, conic, suzanna, premotor, nowlan
Level 49. Words: formally, largely, alon, eralization, backprop, haar
Level 47. Words: believe, invest, distal, stephen, environmental, ically, furnace, aged, straightfor, impossible
Level 48. Words: purely, tinit, resemblance, guaranteed, srihari, dominant, manipulated, discarded, race, entered
Level 49. Words: garcia, incorrectly, capturing, disagree, maintained, smoothed
Level 49. Words: continually, ory, frequent, unchanged, gilbert, variabilities
Level 48. Words: thrust, cockroach, capac, addressing, immune, matlab, lhe, impression
Level 49. Words: formulation, premise, diameter, sand, protable, optimized
Level 49. Words: emphasis, faced, denotes, imple, diate, ent, scann, greenspan
Level 49. Words: calculating, assuump, tani, parenthesis, explained, lithography, associator, summation, mcclelland
Level 48. Words: flipped, adopt, preserving, estimating, technically, struc, approx, decent, flipping, qian
Level 49. Words: rotation, prosodic, scaled, circular, monotonically, displaying, klein, markedly
Level 49. Words: nist, hingedescent, criticism
Level 49. Words: haussler, scenario
Level 49. Words: yin
Level 48. Words: independence, designate, curve, sites, vcdim, untuned, feasible, suppression, opened, ministic
Level 49. Words: nates, putative, jectories, aij, cambridge, mountcastle, monro, remarkable, nnc2, promise
Level 48. Words: development, supervised, persistence, standard, capacitive, putation, ithaca
Level 49. Words: monotonicity, ronny, wes, hodgkin, evidential, bite, coded
Level 49. Words: wisc, albeit, including, overview, homing, potentially, describe, neuronal, bolic, ysis
Level 49. Words: zij
Level 49. Words: tang, aixi, rectangular, frequently, mip, mfc, maximise, converged
Level 49. Words: casting, worth, incorporation, consisting, mathematical, teacher, targeted, divergence
Level 48. Words: identify, reduced, oot, executes, price, dependence, spot, resource, kda, quantified
Level 49. Words: activity, interconnection, slight, relies, remarkably, agrees, written
Level 48. Words: yielded, compositional, econometric, neocognitron, recognise, densities, letter, implementing, sentence, asymptotes
Level 48. Words: yves, saturated, interpreter
Level 49. Words: cord
Level 48. Words: mention, kiang, lesson, manuscript, semantically, conditionally, visual, increas, compensated
Level 49. Words: rli, eleven, faber, organiza, resent, derlying, inversion
Level 49. Words: hessian, magnus
Level 49. Words: hecht, protect, real, lastly, magic, attributed, sufficient, desired, delayed
Level 49. Words: deciding
Level 49. Words: streaming, reversed, conv
Level 47. Words: huffman, sinusoid, medical, aim, michael, borrowed, goal, sensor, molecules, terminated
Level 48. Words: nth, coincide, ramnet, doubly, compromise, polygonal, regarded, surprising, respective, bethesda
Level 49. Words: motoneuron, maturity, occupies, equ, discovering, axon, pole, squaring, ignores
Level 49. Words: complex, modulus, goodwin, barto, generalisation, eliminated, agrees, lisberger, totally, jection
Level 49. Words: discarded, restarting, optimising
Level 49. Words: minus, metamorphosis
Level 48. Words: freitas
Level 49. Words: alon, equate, continous
Level 49. Words: calif, rebound, imaging, analyte
Level 48. Words: original, sition, faithfully, mip, symmetrized
Level 49. Words: glint, teller, totic
Level 49. Words: narendra, storm, emitted, frean, wonder, hidden, onal, sourlas, lazy
Level 49. Words: lecun, understanding
Level 49. Words: const, annotated, dictates
Level 48. Words: acknowledges, williamson, initialisation, decreased, distinguishable, modifies, modified, overlapping, recep, teller
Level 49. Words: sacrificing
Level 49. Words: equation
Level 49. Words: active, walk, edwin, radical
Level 49. Words: circles, esti, figural, premise, crossing, clones, overlaid, haykin, causing, anomalous
Level 49. Words: conditional, umass, mif, tered, midline, positive, liver
Level 49. Words: creasing, kamp, objection, dimensionalities, borisyuk, bivariate, tme, wider, searched, descriptive
Level 49. Words: separating, willing, engineered, strike, boxlet, mechanism, lated
Level 48. Words: anticipate, globin, mathematic, owl, frean, obtaining, iteratively, approximating, jong, misclassified
Level 49. Words: native, trivially, hole, seemore, circumvent, important
Level 48. Words: coder, instruction, sampled
Level 49. Words: simulation, measurable, tly, integrates, art
Level 49. Words: salzberg, decreasing
Level 48. Words: wrt, largest, hillclimbing, try, holden, shadlen, discusses, rendering, presumed, concerned
Level 49. Words: oculocentric, topologically, residue, counterclockwise, triples, replica, edge_x, operational, cliques
Level 49. Words: rein, tsai, representational, driving
Level 48. Words: miyata, weaknesses, rapidly, collapsing, biomedical, inherently, sorting, influences, study, cellular
Level 48. Words: petersen, installed, grey, encounter, quiescent
Level 47. Words: huang, challenging, equidistant, vastly, drifting, lancaster, vas, defect, scored, dimensionalities
Level 48. Words: reflectance, order, skeletal, hancock, guage, sity, independently, isscc, complexes, animal
Level 48. Words: structured, algorithmic, conceptual, plied, fimction, mansour, grouping, benchmarking, unrepresented, lenz
Level 49. Words: indication, explained, anderson, ilwl, reader, manipulator, performer, reliably, captured, built
Level 49. Words: iter, variances, flavor
Level 49. Words: denotes, turing, differentiate
Level 49. Words: assist, option, dolphin, deviation, experiment, imputation, develop, gateway, multistep
Level 49. Words: silicon, cheng
Level 49. Words: nonmonotonic, contradiction
Level 49. Words: multiprocessor
Level 49. Words: formulated
Level 49. Words: longest, walking, principally, scenes, failed
Level 49. Words: multigrid, operated, elimination, extracellular, asme, encoder, onoda, fitzpatrick, sheffield
Level 48. Words: assert, multigrid, merzenich, rgb, dimensionless, reflectance, reflecting, asr, stripes, finding
Level 49. Words: suboptimal, scrambled
Level 49. Words: stant, influenced, women
Level 49. Words: revealing, globin
Level 49. Words: international, prototypes
Level 47. Words: glasses, tempt, stripes, entered, overcomes, briefly, scattered, pricing, acuity, stratum
Level 48. Words: integrating, readable, bruck, pac, coincidence, alternation, pheromone, animation, optimum, suggesting
Level 49. Words: covered, mrf, popular, discriminator, implies, parametrically
Level 49. Words: cybenko
Level 48. Words: mal
Level 49. Words: world, modem
Level 48. Words: commit, jersey, knowing, testable, fourteen, shattering, vicinity, unrealizable, inh, found
Level 49. Words: realizes, concern, susspot
Level 49. Words: formal, tuning, ram, deficit
Level 49. Words: november
Level 48. Words: thresh, kmax
Level 47. Words: chewing, lifetime, perceptron, generative, generator, hammerstrom, welch
Level 48. Words: erkki, arithmetic, adaptability, omitting, heterogeneity, previous, properly, glucose, washington, monolithic
Level 49. Words: recalled, bowl, jour, ca2, cajal, recurrence
Level 49. Words: petersen, imply, ascender, legitimate, aut
Level 49. Words: generalisation, deriv, characterizes
Level 48. Words: encode, cortico, generaliza, wix, statement, freedman, setpoint, calif, want, conceptual
Level 49. Words: dream, unmodified, main
Level 49. Words: institute, exception, efficiently, drifting
Level 49. Words: strategy, atlantic, millisecond, parameterised, irrespective, rectangular
Level 49. Words: weinshall, modifying, quicker
Level 49. Words: clus